IRISA

INSTITUT DE RECHERCHE
EN INFORMATIQUE
ET SYSTÈMES ALÉATOIRES

Campus Universitaire de Beaulieu
35042 RENNES CEDEX FRANCE
Tél.: 02 99 84 71 00 - Télex : UNIRISA 950 473 F
Télécopie : 02 99 84 71 71

Test Specification

6to4 Conformance Test Suite

Technical Document V 3.3

IRISA (Europe)
TAHI Project (Japan)
TTA/IT Testing Laboratory (Korea)

Table Of Contents

1 Introduction

The 6to4 Mechanism is described in RFC 3056 **[RFC3056]** (Connection of IPv6 Domains via IPv4 Clouds, B. Carpenter, K. Moore, February 2001) with references to RFC 2893 **[RFC2893]** (Transition Mechanisms for IPv6 Hosts and Routers R. Gilligan, E. Nordmark, August 2000). The motivation for the 6to4 Mechanism is to allow isolated IPv6 domains or hosts, attached to an IPv4 network which has no native IPv6 support, to communicate with other such IPv6 domains or hosts with minimal manual configuration, before they can obtain native IPv6 connectivity.

An additional RFC in standard track status, extends **[RFC3056].** The operation of 6to4 routers requires either that the routers participate in IPv6 inter-domain routing, or that the routers be provisioned with a default route. The RFC 3068 **[RFC3068]** proposes a standard method to define the default route. It introduces the IANA assigned "6to4 Relay anycast prefix" from which 6to4 packets will be automatically routed to the nearest available router. It allows the managers of the 6to4 relay routers to control the sources authorized to use their resource. It makes it easy to set up a large number of 6to4 relay routers, thus enabling scalability.

This documents gives 6to4 Conformance Test Specifications developed by the IRISA with technical inputs from:

- TAHI Project (Japan)
- TTA/IT Testing Laboratory (Korea)
- UNH InterOperability Lab (USA)

These specifications cover only **[RFC3056]** and does not take into account **[RFC3068].**

Moreover, the RFC 3056**[RFC3056]** presents a model where, instead of static configuration, BGP4+ is used between 6to4 Relay Routers and 6to4 Routers. As described in **[6to4Security]** this model is not known to be used at the time of writing; this is probably caused by the fact that parties that need 6to4 are those that are not able to run BGP, and because setting up these sessions would be much more work for relay operators. Although, this model avoid “trusted relay” and improve security aspects of 6to4, it will not be take into account in this test suite. Indeed, if the 6to4 router established a BGP session between all the possible 6to4 relays, the traffic from non-6to4 sites would always go through "home relay", and configuring "trusted relay" would not be a problem.

[RFC3056] involves three kinds of devices:

- 6to4 router
- 6to4 host
- Relay Router

A Relay Router is a 6to4 router configured to support transit routing between 6to4 addresses and native IPv6 addresses. It means that the only difference between Relay Routers and 6to4 Routers concerns routing policy. Because we will not take into account Routing policies, the following test suite will cover Relay Routers as well as simple 6to4 Routers. Moreover because a 6to4 router acts also as a simple router, we also have to run conformance test suite for core protocol on this router.

A 6to4 host is an IPv6 host, which have at least one 6to4 address. In all other respects it is a standard IPv6 host. Although it is possible to apply the 6to4 mechanism to an individual IPv6 host, **[RFC3056]** does not discuss about this case that raises serious scaling issues. Thus, to check conformance of a 6to4 host, we only need to run conformance test suite for core protocol on this host.

Even though test suite was verified, there can still be a few mistakes. All suggestions are welcome and can be send to:

- Frédéric Roudaut (frederic.roudaut@irisa.fr)
- César Viho (Cesar.Viho@irisa.fr)
- Annie Floch (Annie.floch@irisa.fr)

Test with (*) are only described in the extended version of the 6to4 Conformance Test suite. Their run is only for informational purpose, if needed. Indeed their run does not give more to the test results

2 Test Case Description

Each test case of this 6to4 Conformance Test suite is described using the following sections:

Purpose:	The goal of this section is to describe briefly in a general way the main aim of the test case.
References:	This section is used to give references in the standards concerning the test purpose.
Resource Requirement:	It describes test equipments needed to perform the test case. It can be hardware or software need.
Test Requirement:	It is used to describe specific configurations for the node to test or for the tester in order to perform the test case. This section is in fact the initialization part of the test case.
Discussion:	This section gives more specificaly, the aim of the test according our current test architecture. It can also describe what are the awaited problems …
Packets:	All the different requiered packets for the test case will be describe in this part. XXXXX : the value cannot be predicted. This key-word is only used in packets sent by the NUT. Indeed it is always predictable in packets sent by the tester. This value can also depend from other possible predictable fields and from unpredictable fields. This is the case of IPv4 checksums in packets sent by the NUT while we cannot predict TTL values. Nevertheless, UDP/TCP Checksum may be predicted if we know every field defined in the TCP/UDP pseudo-header and header **[RFC2460] [RFC 793]**. To calculate : The value can be calculated. It depends from other predictable fields. ( ) : The value is not strictely fixed. In packets sent by the tester it is juste an example of a possible value. In packets sent by the NUT, it means that this value is a consequence of a previous possible value chosen by the tester. All the other values are fixed.
Procedure:	The test instructions to carry out the test are described step-by-step in this part.
Observable Results:	This section emphasizes the Observable Results to check by the tester to verify that the NUT is operating as specified in the standards.
Test Sequence:	The test sequence describes packet exchanges at the IP Layer level between entities available in the test case. Packet exchange labeled by a number references the corresponding test intruction of the “Procedure” section. When the Packet Exchange is not labeled, it means that it is an observable result.
Postambule:	This section gives the current state of the Node Under Test after the test execution. This part is used to have the Node Under Test in its initial state after a run of the test case.
Possible Problem:	This section gives a description of possible problems that an execution of the test case could encounter. It discusses know issues which could lead to a “FAIL” verdict.

A few tests of this test suite need to have the capability to modify the different Link MTU. Sometimes, the 6to4 pseudo-interface is initialized with a static value of 1280 Bytes which is the IPv6 Minimimum Link MTU. If the NUT cannot modify this value, these tests MUST be skipped. Such tests are described using the symbol **.

3 Terminology

Acronyms:

The following acronyms will be used in this document:

TR: Test Router.

TN: Test Node.

NUT: Node Under Test.

Terminology:

The terminology defined in **[RFC3056]** applies also to this document:

6to4 pseudo-interface: 6to4 encapsulation of IPv6 packets inside IPv4 packets occurs at a point that is logically equivalent to an IPv6 interface, with the link layer being the IPv4 unicast network. This point is referred to as a pseudo-interface. Some implementors may treat it exactly like any other interface and others may treat it like a tunnel end-point.

6to4 prefix: an IPv6 prefix constructed according to the rule in **Section 2, [RFC 3056]**.

6to4 address: an IPv6 address constructed using a 6to4 prefix.

Native IPv6 address: an IPv6 address constructed using another type of prefix than 6to4.

6to4 router (or 6to4 border router): an IPv6 router supporting a 6to4 pseudo-interface. It is normally the border router between an IPv6 site and a wide-area IPv4 network.

6to4 host: an IPv6 host, which happens to have at least one 6to4 address. In all other respects it is a standard IPv6 host.

6to4 site: a site running IPv6 internally using 6to4 addresses, therefore containing at least one 6to4 host and at least one 6to4 router.

Relay router: a 6to4 router configured to support transit routing between 6to4 addresses and native IPv6 addresses.

6to4 exterior routing domain: a routing domain interconnecting a set of 6to4 routers and relay routers. It is distinct from an IPv6 site's interior routing domain, and distinct from all native IPv6 exterior routing domains.

V4ADDR: globally unique 32-bit IPv4 address allocated to a 6to4 router and which will be used to create the 6to4 Prefix of the network. The corresponding Prefix format can be abbreviated as *2002:V4ADDR::/48.*

4 Topology and Test Configuration

Test Architecture:

The NUT and the tester are connected with 2 links in which no other communications are allowed. The tester uses a packet generator to send IPv6 packets on the links, and a packet analyzer to see the packets sent by the NUT. The logical environment represented by this architecture is shown on Figure 1. All traffic from the NUT, going to the Internet v4 has to be forwarded by TR. It means that all checks for outgoing and incoming packets will have to be done on this router.

Each Link is an RJ45 crossed cable. One of these links represents the IPv4 Link whereas the other is for the IPv6 Link. This topology is the commun topology of all test cases of this 6to4 test suite.

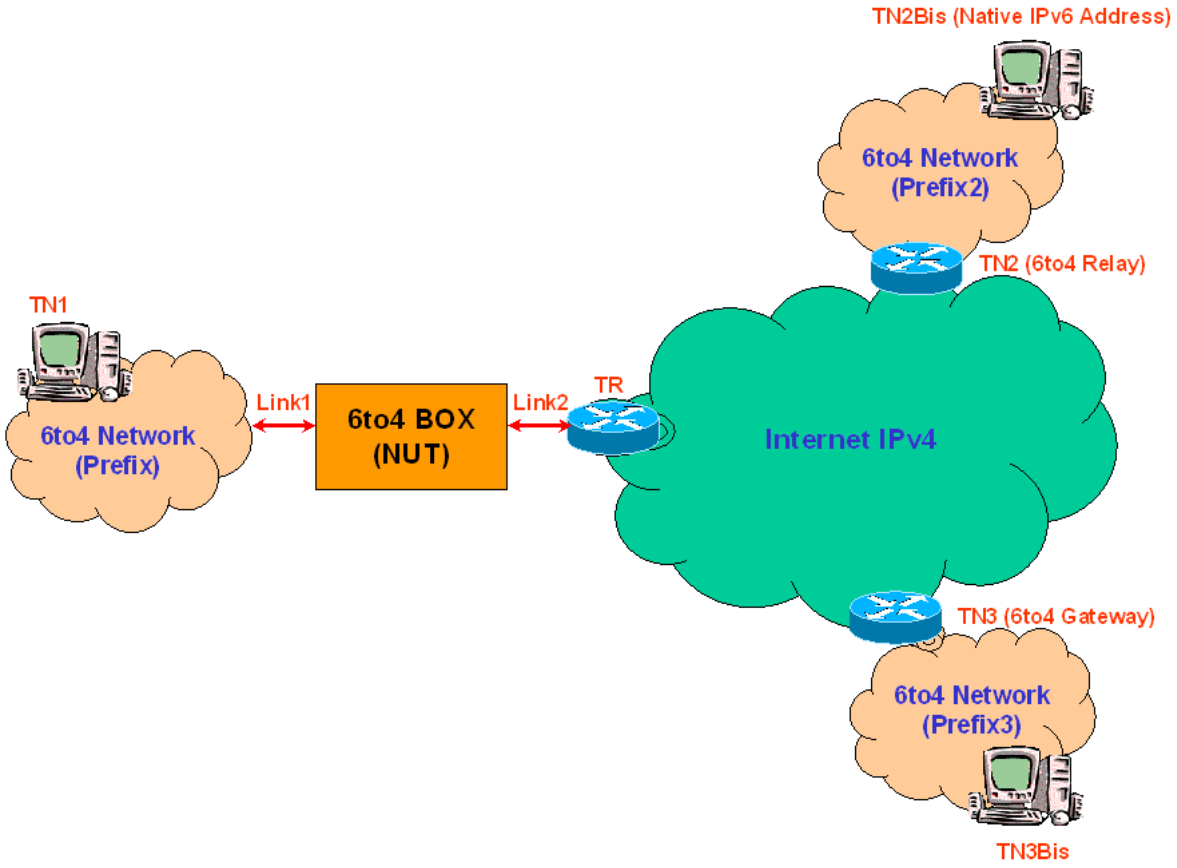


Figure 1: Logical Test Environment for 6to4 Conformance Testing

Addressing:

The following defined values are just an exemple of needed values for this test suite:

- **TR:**

  MAC address: 00:00:00:00:a1:a1

  IPv4 address Link2: 131.254.200.2 netmask 255.255.255.0

- **TN1:**

  MAC address: 00:00:00:00:a0:a0

  Global IPv6 address with 6to4 Prefix: 2002:83fe:c801:0:200:ff:fe00:a0a0

  Link local address: fe80::200:ff:fe00:a0a0

- **TN2:**

IPv4 address: 131.254.201.1 netmask 255.255.255.0

6to4 Prefix2: 2002:83fe:c901::/64

- **TN2Bis:**

  Global IPv6 Native address: 3ffe:501:ffff:100:200:ff:fe00:a2a2

- **TN3:**

  IPv4 address: 131.254.202.1 netmask 255.255.255.0

  6to4 Prefix3: 2002:83fe:ca01::/64

- **TN3Bis:**

  Global IPv6 address with 6to4 Prefix3: 2002:83fe:ca01:0:200:ff:fe00:a3a3

- **6to4 Box:**

  6to4 Prefix: 2002:83fe:c801::/64

  NUT Global IPv6 address Link2 with 6to4 Prefix: 2002:83fe:c801::EUI-64 MAC Link2 (6to4 Address)

  NUT Global IPv6 address Link1 with 6to4 Prefix: 2002:83fe:c801::EUI-64 MAC Link1

  IPv4 address Link2: 131.254.200.1 netmask 255.255.255.0

We need also:

- **Prefix4:** 2002:83fe:cb01:: (with an unreachable IPv4 address: 131.254.203.1)
- **6to4 address with Prefix4:** 2002:83fe:cb01:0:200:ff:fe00:a4a4
- **TN:** 2002:83fe:c801:0:200:ff:fe00:a5a5 (An unreachable host in the 6to4 Network)

To complete, this test topology, we need TN2 as the default 6to4 relay of the NUT and a default IPv4 route for the NUT. It means that on the NUT we should have:

- **default IPv4 Route:** 131.254.200.2 (TR)
- **default IPv6 Route:** 2002:83fe:c901:: (TN2)

On NUT the IPv4 MTU will be set to 1500 when not explicitaly precised. We will consider that medias of the IPv4 clouds are Ethernet Links although this test suite could be run without modification on other 802.X Networks. As defined in **[RFC2464]**, The default MTU size for IPv6 packets on an Ethernet is 1500 octets.

Moreover for best tests analyze, it could be required to desactivate Router Advertisement from the NUT.

Execution of test cases:

Each test case described here should begin when the NUT is at its initial state. We tried to limit the test link effect so that the test cases can be executed in sequence. However, some tests will have more reliable verdicts in a lone run. As no procedure ensures perfect clean-up when a NUT is found to be non conformant, a reinitialization of the NUT should follow every test case that leads to a verdict « FAIL ».

A few tests of this test suite need to have the capability to modify the different Link MTU. Sometimes, the 6to4 pseudo-interface is initialized with a static value of 1280 Bytes which is the IPv6 Minimum Link MTU. If the NUT cannot modify this value, these test MUST be skipped. Such tests are described using the symbol **.

Link-local addresses resolution and ARP:

As part of the Neighbor Discovery protocol **[RFC2461]**, the NUT should send Neighbor Solicitation packets to discover or probe the link-local address of a node at any time. Moreover the Node should too send ARP Packet to discover MAC address From an IPv4 Address. Thus, in most cases, the tester node should respond by a Neighbor Advertisement Packet on the IPv6 Link or an ARP Reply packet on the IPv4 Link.

With, our test topology, it means that the NUT will send Neighbor Solicitation Packet to probe the link-local address of TN1 or to discover its MAC address and ARP Request Message to obtain the MAC address of TR.

These default packets are described hereafter. The values set in these packets should be taken as long as no explicit values are given.

- ARP Request (From NUT to get MAC Address of TR) (length: 28 bytes)

ARP Header (length: 28)
Hardware: 1 Protocol: 2048

HLEN: 6
PLEN: 4
Operation: 1
SenderHAddr: **NUT MAC Address Link2**
SenderPAdd: **NUT IPv4 address Link2**
TargetHAddr: XXXXX
TargetPAddr: **TR IPv4 address Link2**

- ARP Reply (From TR, to give its MAC Address to the NUT) (length: 28 bytes)

ARP Header (length: 28)

Hardware: 1
Protocol: 2048
HLEN: 6
PLEN: 4
Operation: 2
SenderHAddr: **TR MAC Address Link2**
SenderPAdd: **TR IPv4 address Link2**
TargetHAddr: **NUT MAC Address Link2**
TargetPAddr: **NUT IPv4 address Link2**

- Neighbor Solicitation (From NUT to get MAC Address of TN1) (length: 72 bytes)

IPv6 Header (length: 40)

Version: 6
TrafficClass: 0
FlowLabel: 0
PayloadLength: 32
NextHeader: 58
HopLimit : 255
SourceAddress: **NUT IPv6 link-local/Global address**
DestinationAddress: **TN1 IPv6 solicited node multicast address**

Neighbor Solicitation (length: 32)

Type: 135
Code: 0
Checksum: XXXXX
Reserved: 0
TargetAddress: **TN1 Global IPv6 Address with 6to4 Prefix or TN1 Link-local Address**

Option ICMPv6 Source Link Layer (length:8)

Type: 1
Length: 1
LinkLayerAddress: **NUT MAC Address Link1**

- Neighbor Solicitation (From NUT to check reachability of TN1) (length: 72 bytes)

IPv6 Header (length: 40)

Version: 6
TrafficClass: 0
FlowLabel: 0
PayloadLength: 32
NextHeader: 58

HopLimit : 255 SourceAddress: **NUT IPv6 link-local/Global address** DestinationAddress: **TN1 Global IPv6 Address with 6to4 Prefix or TN1 Link-local Address**
Neighbor Solicitation (length: 32) Type: 135 Code: 0 Checksum: XXXXX Reserved: 0 TargetAddress: **TN1 Global IPv6 Address with 6to4 Prefix or TN1 Link-local Address** **Option ICMPv6 Source Link Layer (length:8)** Type: 1 Length: 1 LinkLayerAddress: **NUT MAC Address Link1**

- Neighbor Advertisement (From TN1, to give its MAC Address to the NUT) (length: 72 bytes)

IPv6 Header (length: 40) Version: 6 TrafficClass: 0 FlowLabel: 0 PayloadLength: 32 NextHeader: 58 HopLimit : 255 SourceAddress: **TN1 Global IPv6 Address with 6to4 Prefix or TN1 Link-local Address** DestinationAddress: **NUT IPv6 link-local/Global address (Same as the source field in the corresponding Neighbor Solicitation)**
Neighbor Advertisement (length: 32) Type: 136 Code: 0 Checksum: To calculate Rflag: 0 Sflag: 1 Oflag: 1 Reserved: 0 TargetAddress: **TN1 Global IPv6 address with 6to4 Prefix or TN1 Link-local Address (Same as the Target Address field in the corresponding Neighbor Solicitation)** **Option ICMPv6 Target Link Layer (length:8)** Type: 2 Length: 1 LinkLayerAddress: **TN1 MAC Address**

5 Test suite for 6to4 Routers

5.1 Encapsulation/Decapsulation and Forwarding

Packets leaving a 6to4 network are encapsulated by the 6to4 router before to be forwarded. They are sent either to the corresponding 6to4 router if the destination address is a 6to4 address, either to the default relay router. We remind that we do not take into account the possible use of BGP-4+ between 6to4 Relay Routers and 6to4 Routers. Moreover, tunneled packets coming from another 6to4 relay/router are decapsulated by the 6to4 router before to be submit to the local routing policy.

The goal of this section is to test the basic encapsulation/decapsulation mechanism with ICMPv6 Echo Request Packets. This section wants also to check the correct forwarding to IPv6 native host or to 6to4 hosts.

5.1.1 From IPv6 Network of NUT

5.1.1.1 Basic Encapsulation and forwarding

Purpose:

Check Encapsulation process done by the NUT for packet originating from the 6to4 Network to Native IPv6 Addresses and to other 6to4 Addresses.

References:

- **[RFC3056]** ***Section 3***

Resource Requirement:

- Packet generator
- Monitor To capture Packets:

Test Requirement:

NONE

Discussion:

Packets leaving a 6to4 network are encapsulated by the 6to4 router before to be forwarded. They are sent either to the corresponding 6to4 router if the destination address is a 6to4 address, either to the default relay router. IPv6 Packets are transmitted in IPv4 Packets with an IPv4 protocol type of 41. Thus, Echo Request sent by TN1 to IPv6 Native addresses must be tunneled by the NUT to TN2 and Echo Request sent by TN1 to 6to4 addresses with Prefix3 must be tunneled by the NUT to TN3. Moreover single-hop model is implemented by having the encapsulating and decapsulating nodes process the IPv6 hop limit field as they would if they were forwarding a packet on to any other datalink. That is, they decrement the hop limit by 1 when forwarding an IPv6 packet. (The originating node and final destination do not decrement the hop limit.)

Packets:

- ICMPv6 Echo Request (length: 56 bytes)

IPv6 Header (length: 40)
Version: 6 TrafficClass: (0) FlowLabel: (0) PayloadLength: 16 NextHeader: 58 HopLimit : (64) SourceAddress: **TN1 Global IPv6 Address with 6to4 Prefix** DestinationAddress: **TN2Bis Global IPv6 Native Address or TN3Bis Global IPv6 Address with 6to4 Prefix3**

ICMPv6 Echo Request (length: 16)
Type: 128 Code: 0 Checksum: To calculate Identifier: (512) SequenceNumber: (0) **Payload (length: 8)** Data: (01234567 89abcdef)

- ICMPv6 Echo Request Tunneled (length: 76 bytes)

IPv4 Header (length: 20)
Version: 4 IHL: 5 TypeOfService: 0 TotalLength: 76 Identifier: XXXXX Reserved: 0 DF: 0 MF: 0 FragmentOffset: 0 TTL: XXXXX Protocol: 41 HeaderChecksum: XXXXX SourceAddress: **NUT IPv4 Address Link 2** DestinationAddress: **TN2 IPv4 Address or TN3 IPv4 Address**
Same as the ICMPv6 Echo Request Packet (length: 56) (Except than the Hop Limit is one less. ie set to 63 in our case)

Procedure:

1. TN1 sends an “ICMPv6 Echo Request” to the NUT for TN2bis (Native IPv6 Address). The IPv6 Destination Address is TN2Bis Global IPv6 Native Address.
2. TN1 sends an “ICMPv6 Echo Request” to the NUT for TN3bis (6to4 Address with Prefix3). The IPv6 Destination Address is TN3Bis Global IPv6 Address with 6to4 Prefix3.

Observable Results:

- **Step 1:** The NUT tunnels this “ICMPv6 Echo Request” to TN2 in an IPv4 Packet “ICMPv6 Echo Request Tunneled”. The IPv4 Destination Address is TN2 IPv4 Address and the IPv6 Destination Address is TN2Bis Global IPv6 Native Address.
- **Step 2:** The NUT tunnels this “ICMPv6 Echo Request” to TN3 in an IPv4 Packet “ICMPv6 Echo Request Tunneled”. The IPv4 Destination Address is TN3 IPv4 Address and the IPv6 Destination Address is TN3Bis Global IPv6 Address with 6to4 Prefix3.

Test Sequence:

Tester	Link1 [IPv6]	RUT	Link2 [IPv4]	Tester(IPv6)
	ICMPv6 Echo Request			
TN1	---------->	1		
			ICMPv6 Echo Request Tunneled	
		Step 1	---------->	TN2(TN2Bis)
	ICMPv6 Echo Request			
TN1	---------->	2		
			ICMPv6 Echo Request Tunneled	
		Step 2	---------->	TN3(TN3Bis)

Postambule:

A successful run of this test case should have only modified the NDP and ARP caches. Thus, to put the NUT in its initial state they have to be cleaned.

After the test execution, their entry will be the following:

ARP Cache	NDP Cache
TR	TN1

5.1.1.2 Basic Encapsulation of IPv6/UDP Messages (*)

Purpose:

Check Encapsulation of IPv6/UDP Messages. IPv6/UDP Message is sent from The 6to4 Network to one Native IPv6 Address.

References:

- **[RFC3056]** ***Section 3***

Resource Requirement:

- Packet generator
- Monitor To capture Packets:

Test Requirement:

NONE

Discussion:

IPv6/UDP Packets sent by TN1 to IPv6 Native addresses must be tunneled by the NUT to TN2, that is the default 6to4 relay of the NUT. This test adds nothing really more to the previous one. It just shows the correct encapsulation and forwarding of UDP packets.

Packets:

- IPv6/UDP (length: 56 bytes)

IPv6 Header (length: 40)
Version: 6 TrafficClass: (0) FlowLabel: (0) PayloadLength: 16 NextHeader: 17 HopLimit : (64) SourceAddress: **TN1 Global IPv6 Address with 6to4 Prefix** DestinationAddress: **TN2Bis Global IPv6 Native Address**
UDP (length: 16) SourcePort: (1000) DestinationPort: (1000) Length: 16 Checksum: To calculate **Payload (length: 8)** Data: (01234567 89abcdef)

- IPv6/UDP Tunneled (length: 76 bytes)

IPv4 Header (length: 20)
Version: 4 IHL: 5 TypeOfService: 0 TotalLength: 76 Identifier: XXXXX Reserved: 0 DF: 0 MF: 0 FragmentOffset: 0 TTL: XXXXX Protocol: 41

HeaderChecksum: XXXXX SourceAddress: **NUT IPv4 Address Link 2** DestinationAddress: **TN2 IPv4 Address**
Same as IPv6/UDP Packet (length: 56) **(Except than the Hop Limit is one less. ie set to 63 in our case)**

Procedure:

1. TN1 sends an “IPv6/UDP” packet to the NUT for TN2bis.

Observable Results:

- **Step 1:** The NUT tunnels this “IPv6/UDP” packet to TN2 in an IPv4 Packet “IPv6/UDP Tunneled”.

Test Sequence:

Tester	Link1 [IPv6]	RUT	Link2 [IPv4]	Tester(IPv6)
	IPv6/UDP			
TN1	---------->	1		
			IPv6/UDP Tunneled	
		Step 1	---------->	TN2(TN2Bis)

Postambule:

A successful run of this test case should have only modified the NDP and ARP caches. Thus, to put the NUT in its initial state they have to be cleaned.

After the test execution, their entry will be the following:

ARP Cache	NDP Cache
TR	TN1

5.1.1.3 Basic Encapsulation of IPv6/TCP Messages (*)

Purpose:

Check Encapsulation of IPv6/TCP Messages. IPv6/TCP Message is sent from the 6to4 network to one Native IPv6 Address.

References:

- **[RFC3056]** ***Section 3***

Resource Requirement:

- Packet generator
- Monitor To capture Packets:

Test Requirement:

NONE

Discussion:

IPv6/TCP Packets sent by TN1 to IPv6 Native addresses must be tunneled by the NUT to TN2, that is the default 6to4 relay of the NUT. This test adds nothing really more to the previous one. It just shows the correct encapsulation and forwarding of TCP packets.

Packets:

- IPv6/TCP (length: 68 bytes)

IPv6 Header (length: 40) Version: 6 TrafficClass: (0) FlowLabel: (0) PayloadLength: 28 NextHeader: 6 HopLimit : (64) SourceAddress: **TN1 Global IPv6 Address with 6to4 Prefix** DestinationAddress: **TN2Bis Global IPv6 Native Address**
TCP (length: 28) SourcePort: (1000) DestinationPort: (1000) SequenceNumber: (0) AcknowledgmentNumber: (0) DataOffset: (5) Reserverd: (0) URGFlag: (0) ACKFlag: (0) PSHFlag: (0) RSTFlag: (0) SYNFlag: (1) FINFlag: (0) Window: (0) Checksum: To calculate UrgentPointer: (0) **Payload (length: 8)** Data: (01234567 89abcdef)

- IPv6/TCP Tunneled (length: 88 bytes)

IPv4 Header (length: 20) Version: 4 IHL: 5 TypeOfService: 0 TotalLength: 88 Identifier: XXXXX Reserved: 0 DF: 0 MF: 0 FragmentOffset: 0 TTL: XXXXX Protocol: 41 HeaderChecksum: XXXXX SourceAddress: **NUT IPv4 Address Link 2** DestinationAddress: **TN2 IPv4 Address**
Same as IPv6/TCP Packet (length: 68) **(Except than the Hop Limit is one less. ie set to 63 in our case)**

Procedure:

1. TN1 sends an “IPv6/TCP” packet to the NUT for TN2bis.

Observable Results:

- **Step 1:** The NUT tunnels this “IPv6/TCP” packet to TN2 in an IPv4 Packet “IPv6/TCP Tunneled”.

Test Sequence:

Tester	Link1 [IPv6]	RUT	Link2 [IPv4]	Tester(IPv6)
	IPv6/TCP			
TN1	---------->	1		
			IPv6/TCP tunneled	
		Step 1	---------->	TN2(TN2Bis)

Postambule:

A successful run of this test case should have only modified the NDP and ARP caches. Thus, to put the NUT in its initial state they have to be cleaned.

After the test execution, their entry will be the following:

ARP Cache	NDP Cache
TR	TN1

5.1.2 To IPv6 Network of NUT

5.1.2.1 Basic Decapsulation and Forwarding

Purpose:

Check Decapsulation of Echo Request Messages. ICMPv6 Echo Request are sent from one Native IPv6 Address and from another 6to4 Address to an host located in the 6to4 network. ICMPv6 Echo Request packets are also sent from one Native IPv6 Address and from another 6to4 Address to the NUT.

References:

- **[RFC3056]** ***Section 5.3***

Resource Requirement:

- Packet generator
- Monitor To capture Packets:

Test Requirement:

NONE

Discussion:

ICMPv6 Echo Request sent by IPv6 Native addresses to TN1 or to the NUT must be tunneled by the nearest 6to4 relay of this host (TN2). ICMPv6 Echo Request sent by 6to4 addresses with Prefix3 to TN1 or to the NUT must be tunneled by the corresponding 6to4 router (TN3) of this host. The single-hop model is implemented by having the encapsulating and decapsulating nodes process the IPv6 hop limit field as they would if they were forwarding a packet on to any other datalink. That is, they decrement the hop limit by 1 when forwarding an IPv6 packet.

Packets:

- ICMPv6 Echo Request (length: 56 bytes)

IPv6 Header (length: 40) Version: 6 TrafficClass: (0) FlowLabel: (0) PayloadLength: 16 NextHeader: 58 HopLimit : (63) SourceAddress: **TN2Bis Global IPv6 Native Address or TN3Bis Global IPv6 Address with 6to4 Prefix3** DestinationAddress: **TN1 Global IPv6 Address with 6to4 Prefix or NUT Global IPv6 Address Link1/Link2 with 6to4 Prefix**
ICMPv6 Echo Request (length: 16) Type: 128 Code: 0 Checksum: To calculate Identifier: (514) SequenceNumber: (0) **Payload (length: 8)** Data: (01234567 89abcdef)

- ICMPv6 Echo Request Tunneled (length: 76 bytes)

IPv4 Header (length: 20)

Version: 4 IHL: 5 TypeOfService: 0 TotalLength: 76 Identifier: XXXXX Reserved: 0 DF: 0 MF: 0 FragmentOffset: 0 TTL: XXXXX Protocol: 41 HeaderChecksum: XXXXX SourceAddress: **TN2 IPv4 Address or TN3 IPv4 Address** DestinationAddress: **NUT IPv4 Address Link 2**
Same as ICMPv6 Echo Request Packet (length: 56) (Except than the Hop Limit is one more. Ie set to 64 in our case)

- ICMPv6 Echo Reply (length: 56 bytes)

IPv6 Header (length: 40) Version: 6 TrafficClass: (0) FlowLabel: (0) PayloadLength: 16 NextHeader: 58 HopLimit : (64) SourceAddress: **NUT Global IPv6 Address Link1/Link2 with 6to4 Prefix** DestinationAddress: **TN2Bis Global IPv6 Native Address or TN3Bis Global IPv6 Address with 6to4 Prefix3**
ICMPv6 Echo Reply (length: 16) Type: 129 Code: 0 Checksum: To calculate Identifier: (514 Same as the ICMPv6 Echo Request) SequenceNumber: (0 Same as the ICMPv6 Echo Request) **Payload (length: 8)** Data: (01234567 89abcdef Same as the ICMPv6 Echo Request)

- ICMPv6 Echo Reply Tunneled (length: 76 bytes)

IPv4 Header (length: 20) Version: 4 IHL: 5 TypeOfService: 0 TotalLength: 76 Identifier: XXXXX Reserved: 0 DF: 0 MF: 0 FragmentOffset: 0 TTL: XXXXX

Protocol: 41 HeaderChecksum: XXXXX SourceAddress: **NUT IPv4 Address Link 2** DestinationAddress: **TN2 IPv4 Address or TN3 IPv4 Address**
Same as ICMPv6 Echo Reply Packet (length: 56) **(Except than the Hop Limit is one less. ie set to 64 in our case)**

Procedure:

1. TN2 sends an "ICMPv6 Echo Request tunneled" from TN2Bis to the NUT for TN1. The IPv4 Source Address is TN2 IPv4 Address and the IPv6 Source Address is TN2Bis Global IPv6 Native Address. Moreover the IPv6 Destination Address is TN1 Global IPv6 Address with 6to4 Prefix.
2. TN3 sends an "ICMPv6 Echo Request tunneled" from TN3Bis to the NUT for TN1. The IPv4 Source Address is TN3 IPv4 Address and the IPv6 Source Address is TN3Bis Global IPv6 Address with 6to4 Prefix3. Moreover the IPv6 Destination Address is TN1 Global IPv6 Address with 6to4 Prefix.
3. TN2 sends an "ICMPv6 Echo Request tunneled" from TN2Bis to the NUT for the NUT itself. The IPv4 Source Address is TN2 IPv4 Address and the IPv6 Source Address is TN2Bis Global IPv6 Native Address. Moreover the IPv6 Destination Address is the NUT Global IPv6 Address Link2 with 6to4 Prefix.
4. TN2 sends an "ICMPv6 Echo Request tunneled" from TN2Bis to the NUT for the NUT itself. The IPv4 Source Address is TN2 IPv4 Address and the IPv6 Source Address is TN2Bis Global IPv6 Native Address. Moreover the IPv6 Destination Address is the NUT Global IPv6 Address Link1 with 6to4 Prefix.
5. TN3 sends an "ICMPv6 Echo Request tunneled" from TN3Bis to the NUT for the NUT itself. The IPv4 Source Address is TN3 IPv4 Address and the IPv6 Source Address is TN3Bis Global IPv6 Address with 6to4 Prefix3. Moreover the IPv6 Destination Address is the NUT Global IPv6 Address Link2 with 6to4 Prefix.
6. TN3 sends an "ICMPv6 Echo Request tunneled" from TN3Bis to the NUT for the NUT itself. The IPv4 Source Address is TN3 IPv4 Address and the IPv6 Source Address is TN3Bis Global IPv6 Address with 6to4 Prefix3. Moreover the IPv6 Destination Address is the NUT Global IPv6 Address Link1 with 6to4 Prefix.

Observable Results:

- **Step 1:** The NUT desencapsulates this "ICMPv6 Echo Request tunneled" and forwards an "ICMPv6 Echo Request" to TN1. The IPv6 Source Address is TN2Bis Global IPv6 Native Address and the IPv6 Destination Address is TN1 Global IPv6 Address with 6to4 Prefix.
- **Step 2:** The NUT desencapsulates this "ICMPv6 Echo Request tunneled" and forwards an "ICMPv6 Echo Request" to TN1. The IPv6 Source Address is TN3Bis Global IPv6 Address with 6to4 Prefix3 and the IPv6 Destination Address is TN1 Global IPv6 Address with 6to4 Prefix.
- **Step 3:** The NUT sends an "ICMPv6 Echo Reply tunneled" to TN2 for TN2Bis. The IPv4 Destination Address is TN2 IPv4 Address and the IPv6 Destination Address is TN2Bis Global IPv6 Native Address. Moreover the IPv6 Source Address is the NUT Global IPv6 Address Link2 with 6to4 Prefix.
- **Step 4:** The NUT sends an "ICMPv6 Echo Reply tunneled" to TN2 for TN2Bis. The IPv4 Destination Address is TN2 IPv4 Address and the IPv6 Destination Address is TN2Bis Global IPv6 Native Address. Moreover the IPv6 Source Address is the NUT Global IPv6 Address Link1 with 6to4 Prefix.
- **Step 5:** The NUT sends an "ICMPv6 Echo Reply tunneled" to TN3 for TN3Bis. The IPv4 Destination Address is TN3 IPv4 Address and the IPv6 Destination Address is TN3Bis Global IPv6 Address with 6to4 Prefix3. Moreover the IPv6 Source Address is the NUT Global IPv6 Address Link2 with 6to4 Prefix.
- **Step 6:** The NUT sends an "ICMPv6 Echo Reply tunneled" to TN3 for TN3Bis. The IPv4 Destination Address is TN3 IPv4 Address and the IPv6 Destination Address is TN3Bis Global IPv6 Address with 6to4 Prefix3. Moreover the IPv6 Source Address is the NUT Global IPv6 Address Link1 with 6to4 Prefix.

Test Sequence:

Tester	Link1 [IPv6]	RUT	Link2 [IPv4]	Tester(IPv6)
		1	ICMPv6 Echo Request Tunneled <----------	TN2(TN2Bis)
TN1	ICMPv6 Echo Request <----------	Step 1		
		2	ICMPv6 Echo Request Tunneled <----------	TN3(TN3Bis)
TN1	ICMPv6 Echo Request <----------	Step 2		
		3	ICMPv6 Echo Request Tunneled <----------	TN2(TN2Bis)
		Step 3	ICMPv6 Echo Reply Tunneled ---------->	TN2(TN2Bis)
		4	ICMPv6 Echo Request Tunneled <----------	TN2(TN2Bis)
		Step 4	ICMPv6 Echo Reply Tunneled ---------->	TN2(TN2Bis)
		5	ICMPv6 Echo Request Tunneled <----------	TN3(TN3Bis)
		Step 5	ICMPv6 Echo Reply Tunneled ---------->	TN3(TN3Bis)
		6	ICMPv6 Echo Request Tunneled <----------	TN3(TN3Bis)
		Step 6	ICMPv6 Echo Reply Tunneled ---------->	TN3(TN3Bis)

Postambule:

A successful run of this test case should have only modified the NDP and ARP caches. Thus, to put the NUT in its initial state they have to be cleaned.

After the test execution, their entry will be the following:

ARP Cache	NDP Cache
TR	TN1

5.1.2.2 Basic Decapsulation of IPv6/UDP Messages (*)

Purpose:

Check Decapsulation of IPv6/UDP Messages. IPv6/UDP Message is sent from Native IPv6 Address to one host located on the 6to4 network.

References:

- **[RFC3056]** ***Section 3, 5.3***

Resource Requirement:

- Packet generator
- Monitor To capture Packets:

Test Requirement:

NONE

Discussion:

Packets sent by IPv6 Native addresses to TN1 must be tunneled by the nearest 6to4 relay of this host. TN2 will be this default 6to4 relay. This test adds nothing really more to the previous one. It just shows the correct decapsulation and forwarding of UDP packets.

Packets:

- IPv6/UDP (length: 56 bytes)

IPv6 Header (length: 40)
Version: 6 TrafficClass: (0) FlowLabel: (0) PayloadLength: 16 NextHeader: 17 HopLimit : (63) SourceAddress: **TN2Bis Global IPv6 Native Address** DestinationAddress: **TN1 Global IPv6 Address with 6to4 Prefix**
UDP (length: 16) SourcePort: (1000) DestinationPort: (1000) Length: 16 Checksum: To calculate **Payload (length: 8)** Data: (01234567 89abcdef)

- IPv6/UDP Tunneled (length: 76 bytes)

IPv4 Header (length: 20)
Version: 4 IHL: 5 TypeOfService: (0) TotalLength: 76 Identifier: (515) Reserved: 0 DF: 0 MF: 0 FragmentOffset: 0 TTL: (64)

Protocol: 41 HeaderChecksum: To calculate SourceAddress: **NUT IPv4 Address Link 2** DestinationAddress: **TN2 IPv4 Address**
Same as IPv6/UDP Packet (length: 56) **(Except than the Hop Limit is one more. ie set to 64 in our case)**

Procedure:

1. TN2 sends an “IPv6/UDP tunneled” Packet from TN2Bis to the NUT for TN1

Observable Results:

- **Step 1:** The NUT desencapsulates this “IPv6/UDP tunneled” Packet to TN1 in an IPv6 Packet “IPv6/UDP Packet” and forwards it to TN1.

Test Sequence:

Tester	Link1 [IPv6]	RUT	Link2 [IPv4]	Tester(IPv6)
			IPv6/UDP	
		1	<----------	TN2(TN2Bis)
	IPv6/UDP Tunneled			
TN1	<----------	Step 1		

Postambule:

A successful run of this test case should have only modified the NDP and ARP caches. Thus, to put the NUT in its initial state they have to be cleaned.

After the test execution, their entry will be the following:

ARP Cache	NDP Cache
(TR)	TN1

The ARP Cache may be empty, because no stream has been generated from the 6to4 Network to the IPv4 Network. Nevertheless, we have to keep in mind that TR is the default IPv4 route of the NUT. Thus, it will not be problematic to find TR in the ARP cache...

5.1.2.3 Basic Decapsulation of IPv6/TCP Messages (*)

Purpose:

Check Decapsulation of IPv6/TCP Messages. IPv6/TCP tunneled Message is sent from Native IPv6 Address to one host located on the 6to4 network.

References:

- **[RFC3056]** ***Section 3, 5.3***

Resource Requirement:

- Packet generator
- Monitor To capture Packets:

Test Requirement:

NONE

Discussion:

Packets sent by IPv6 Native addresses to TN1 must be tunneled by the nearest 6to4 relay of this host. TN2 will be this default 6to4 relay. This test adds nothing really more to the previous one. It just shows the correct decapsulation and forwarding of TCP packets.

Packets:

- IPv6/TCP (length: 68 bytes)

IPv6 Header (length: 40)
Version: 6 TrafficClass: (0) FlowLabel: (0) PayloadLength: 28 NextHeader: 6 HopLimit : (63) SourceAddress: **TN2Bis Global IPv6 Native Address** DestinationAddress: **TN1 Global IPv6 Address with 6to4 Prefix**
TCP (length: 28) SourcePort: (1000) DestinationPort: (1000) SequenceNumber: (0) AcknowledgmentNumber: (0) DataOffset: (5) Reserverd: (0) URGFlag: (0) ACKFlag: (0) PSHFlag: (0) RSTFlag: (0) SYNFlag: (1) FINFlag: (0) Window: (0) Checksum: To calculate UrgentPointer: (0) **Payload (length: 8)** Data: (01234567 89abcdef)

- IPv6/TCP Tunneled (length: 88 bytes)

IPv4 Header (length: 20)

Version: 4 IHL: 5 TypeOfService: 0 TotalLength: 88 Identifier: (516) Reserved: 0 DF: 0 MF: 0 FragmentOffset: 0 TTL: (64) Protocol: 41 HeaderChecksum: To calculate SourceAddress: **NUT IPv4 Address Link 2** DestinationAddress: **TN2 IPv4 Address**
Same as IPv6/TCP Packet (length: 68) **(Except than the Hop Limit is one less. ie set to 64 in our case)**

Procedure:

1. TN2 sends an “IPv6/TCP tunneled” Packet from TN2Bis to the NUT for TN1

Observable Results:

- **Step 1:** The NUT desencapsulates this “IPv6/TCP tunneled” Packet to TN1 in an IPv6 Packet “IPv6/TCP Packet” and forwards it to TN1.

Test Sequence:

Tester	Link1 [IPv6]	RUT	Link2 [IPv4]	Tester(IPv6)
			IPv6/TCP	
		1	<----------	TN2(TN2Bis)
	IPv6/TCP Tunneled			
TN1	<----------	Step 1		

Postambule:

A successful run of this test case should have only modified the NDP and ARP caches. Thus, to put the NUT in its initial state they have to be cleaned.

After the test execution, their entry will be the following:

ARP Cache	NDP Cache
(TR)	TN1

The ARP Cache may be empty, because no stream has been generated from the 6to4 Network to the IPv4 Network. Nevertheless, we have to keep in mind that TR is the default IPv4 route of the NUT. Thus, it will not be problematic to find TR in the ARP cache...

5.2 ICMPv6 Error Generation

Since the 6to4 router is a router, it SHOULD be able to correctly generate ICMPv6 error Packets.

The goal of this section is to check that packets which leads to ICMPv6 error messages are discarded and the ICMPv6 Error Packets are correctly sent.

The ICMPv6 Error Messages are:

2. Destination Unreachable
3. Packet Too Big
4. Time Exceeded
5. Parameter Problem

5.2.1 To IPv6 Network of NUT

5.2.1.1 ICMPv6 Destination Unreachable

Purpose:

Check that the NUT properly generates an ICMPv6 Destination Unreachable message

References:

- **[RFC2463]** ***Section 3.1***

Resource Requirement:

- Packet generator
- Monitor To capture Packets:

Test Requirement:

- Remove the Default IPv6 and IPv4 Routes on the NUT.
- The NUT MUST not be listening on port 7777

Discussion:

An ICMPv6 Destination Unreachable Message SHOULD be generated by a router in the originating node in response to a packet that cannot be delivered to a destination for reasons other than congestion.

The ICMPv6 Code may be set to:

- 0 - no route to destination
- 1 - communication with destination administratively prohibited
- 3 - address unreachable
- 4 - port unreachable

Packets:

- ICMPv6 Echo Request (length: 56 bytes)

IPv6 Header (length: 40)

Version: 6
TrafficClass: (0)
FlowLabel: (0)
PayloadLength: 16
NextHeader: 58
HopLimit : (64)
SourceAddress: **TN1 Global IPv6 Address with 6to4 Prefix**
DestinationAddress: **TN2Bis Global IPv6 Native Address** or

TN3Bis Global IPv6 Address with 6to4 Prefix3
ICMPv6 Echo Request (length: 16) Type: 128 Code: 0 Checksum: To calculate Identifier: (612) SequenceNumber: (0) **Payload (length: 8)** Data: (01234567 89abcdef)

- IPv6/UDP (length: 56 bytes)

IPv6 Header (length: 40) Version: 6 TrafficClass: (0) FlowLabel: (0) PayloadLength: 16 NextHeader: 17 HopLimit : (64) SourceAddress: **TN1 Global IPv6 Address with 6to4 Prefix** DestinationAddress: **TN2Bis Global IPv6 Native Address**
UDP (length: 16) SourcePort: (1000) DestinationPort: **7777** Length: 16 Checksum: To calculate **Payload (length: 8)** Data: (01234567 89abcdef)

- ICMPv6 Destination Unreachable (length: 104 bytes)

IPv6 Header (length: 40) Version: 6 TrafficClass: 0 FlowLabel: 0 PayloadLength: 64 NextHeader: 58 HopLimit : XXXXX SourceAddress: **NUT Global IPv6 Address (Link1 or Link2)/NUT Link-local IPv6 address Link1** DestinationAddress: **TN1 Global IPv6 Address with 6to4 Prefix**
ICMPv6 Destination Unreachable (length: 64) Type: 1 Code: **CODE** Checksum: XXXXX Unused: 0 **Payload (length: 56)** Data: the corresponding offending packet (ICMPv6 Echo

Request or IPv6/UDP)

Procedure:

0. Remove the Default IPv6 and IPv4 Routes on the NUT.
1. TN1 sends an “ICMPv6 Echo Request” for TN2Bis (Native IPv6 Address).
2. TN1 sends an “ICMPv6 Echo Request” for TN3Bis (6to4 Address with Prefix3).
3. TN1 sends an “ICMPv6 Echo Request” for TN (Unreachable Host located in the 6to4 Network) to the NUT. I.e the Destination Mac Address is the NUT MAC Address).
4. TN1 sends some “IPv6/UDP” Packet to the NUT (The UDP Port is set to 7777 and the NUT MUST not be listening on this port). The “IPv6/UDP” packets use the following addresses:
    i. The Source Address is **TN1 Link-local IPv6 Address**. The Destination Address is **NUT Link-local IPv6 Address Link1.**
    ii. The Source Address is **TN1 Link-local IPv6 Address**. The Destination Address is **NUT Global IPv6 Address Link1.**
    iii. The Source Address is **TN1 Global IPv6 Address with 6to4 Prefix**. The Destination Address is **NUT Link-local IPv6 Address Link1.**
    iv. The Source Address is **TN1 Global IPv6 Address with 6to4 Prefix**. The Destination Address is **NUT Global IPv6 Address Link1.**
    v. The Source Address is **TN1 Global IPv6 Address with 6to4 Prefix**. The Destination Address is **NUT Global IPv6 Address Link2.**

Observable Results:

- **Step 1:** The NUT replies with an “ICMPv6 Destination Unreachable” to TN1. The source address should be the **NUT Global IPv6 Address Link1 or Link2** or the **NUT Link-local IPv6 address Link1.** The Code Field (**CODE**) SHOULD be set to 0. The offending packet included in the “ ICMPv6 Destination Unreachable ” should be the one which causes the error (ie the “ICMPv6 Echo Request”).

  Remarque: If a default IPv4 Route exists but no default IPv6 route is available in the routing table of the NUT, this “ICMPv6 Destination Unreachable” should also be generated.

- **Step 2:** The NUT replies with an “ICMPv6 Destination Unreachable” to TN1. The source address should be the **NUT Global IPv6 Address Link1 or Link2** or the **NUT Link-local IPv6 address Link1.** The Code Field (**CODE**) SHOULD be set to 3. The offending packet included in the “ ICMPv6 Destination Unreachable ” should be the one which causes the error (ie the “ICMPv6 Echo Request”).

- **Step 3:** The NUT may send Neighbor solicitation Packet on the 6to4 Network for getting MAC Address of TN. However TN will not respond. The NUT replies with an “ICMPv6 Destination Unreachable” to TN1. The source address should be the **NUT Global IPv6 Address Link1** or the **NUT Link-local IPv6 address Link1.** The Code Field (**CODE**) SHOULD be set to 3. The offending packet included in the “ ICMPv6 Destination Unreachable ” should be the one which causes the error (ie the “ICMPv6 Echo Request”). Moreover the NUT MAY send an ICMPv6 Redirect packet in order to tell to TN1 that the Node is on its link.

- **Step 4:** The NUT replies with an “ICMPv6 Destination Unreachable” to TN1 to each packet. The source addresses should be the the same as the destination addresses of the offending packets. The Code Field (**CODE**) SHOULD be set to 4. The offending packet included in the “ICMPv6 Destination Unreachable” should be the one which causes the error (ie the “IPv6/UDP”). The source addresses are:
    i. The **NUT Link-local IPv6 Address Link1.**
    ii. The **NUT Global IPv6 Address Link1.**
    iii. The **NUT Link-local IPv6 Address Link1.**
    iv. The **NUT Global IPv6 Address Link1.**
    v. The **NUT Global IPv6 Address Link2.**

Test Sequence:

Tester	Link1 [IPv6]	RUT	Link2 [IPv4]	Tester(IPv6)
TN1	ICMPv6 Echo Request to TN2Bis ---------->	1		
TN1	<---------- ICMPv6 Destination Unreachable	Step 1		
TN1	ICMPv6 Echo Request to TN3Bis ---------->	2		
TN1	<---------- ICMPv6 Destination Unreachable	Step 2		
TN1	ICMPv6 Echo Request to TN ---------->	3		
TN	<---------- Neighbor Solicitation			
TN1	<---------- ICMPv6 Destination Unreachable	Step 3		
TN1	IPv6/UDP to NUT Link-Local IPv6 Address Link1 ---------->	4i		
TN1	<---------- ICMPv6 Destination Unreachable	Step 4i		
TN1	IPv6/UDP to NUT Global IPv6 Address Link1 ---------->	4ii		
TN1	<---------- ICMPv6 Destination Unreachable	Step 4ii		
TN1	IPv6/UDP to NUT Link-Local IPv6 Address Link1 ---------->	4iii		
TN1	<---------- ICMPv6 Destination Unreachable	Step 4iii		
TN1	IPv6/UDP to NUT Global IPv6 Address Link1 ---------->	4iv		
TN1	<---------- ICMPv6 Destination Unreachable	Step 4iv		
TN1	IPv6/UDP to NUT Global IPv6 Address Link2 ---------->	4v		
TN1	<---------- ICMPv6 Destination Unreachable	Step 4v		

Postambule:

A successful run of this test case should have only modified the NDP cache. Thus, to put the NUT in its initial state it has to be cleaned.

After the test execution, the cache entries entry will be the following:

ARP Cache	NDP Cache
	TN1

After this test you need to set the previous IPv4 and IPv6 routes.

Possible Problem:

- Possible problems can appear in this test case. The “ICMPv6 Destination Unreachable” packet may not match the awaited packet. According to **[RFC2463]**, when processing ICMPv6 messages, Implementations MUST observe the following rule: “every ICMPv6 error message (type < 128) MUST includes as much of the IPv6 offending (invoking) packet (the packet that caused the error) as will fit without making the error message packet exceed the minimum IPv6 MTU”. However, with the full offending packet, the “ICMPv6 Destination Unreachable” packet length is 104 bytes which is less than the minimum IPv6 MTU 1280 bytes.

5.2.1.2 ICMPv6 Packet Too Big(1)

Purpose:

Set the IPv4 MTU on NUT to 1300. As a consequence the tunnel MTU should be 1280. If the NUT receives an IPv6 Packet greater than 1280 bytes, it MUST send an ICMPv6 Packet too big Message with MTU = 1280 and drop the previous packet.

References:

- **[RFC2893]** ***Section 3.2:***

  "**if** (IPv4 path MTU - 20) is less than or equal to 1280

  if packet is larger than 1280 bytes

  Send IPv6 ICMP "packet too big" with MTU = 1280.

  Drop packet.

  else

  Encapsulate but do not set the Don't Fragment flag in the IPv4 header. The resulting IPv4 packet might be fragmented by the IPv4 layer on the encapsulating node or by some router along the IPv4 path.

  endif

  else

  if packet is larger than (IPv4 path MTU - 20)

  Send IPv6 ICMP "packet too big" with

  MTU = (IPv4 path MTU - 20).

  Drop packet.

  else

  Encapsulate and set the Don't Fragment flag in the IPv4 header.

  endif

  endif"

Resource Requirement:

- Packet generator
- Monitor To capture Packets:

Test Requirement:

- The IPv4 MTU on NUT has to be set to 1300. As a consequence the tunnel MTU should be 1280. If the NUT is not able to modify the MTU, It may be done using an ICMP Destination Unreachable message sent from TR to NUT as defined in the PATH MTU Discovery specification [RFC1191].

Discussion:

IPv6/UDP Packets sent by TN1 to IPv6 Native addresses must be tunneled by the NUT to TN2. The IPv4 MTU on NUT is set to 1300. If the NUT receives an IPv6 Packet from TN1 with at least 1281 bytes, it MUST send an ICMPv6 Packet too big Message with MTU = 1280 and drop the previous packet.

Packets:

- IPv6/UDP (length: 1281 bytes)

IPv6 Header (length: 40)
Version: 6 TrafficClass: (0) FlowLabel: (0) PayloadLength: 1241 NextHeader: 17 HopLimit : (64)

SourceAddress: **TN1 Global IPv6 Address with 6to4 Prefix** DestinationAddress: **TN2Bis Global IPv6 Native Address**
UDP (length: 1241) SourcePort: (1000) DestinationPort: (1000) Length: 1241 Checksum: To calculate **Payload (length: 1233)** Data: 1233 Bytes of data

- ICMPv6 Packet Too Big (length: 1280 bytes)

IPv6 Header (length: 40) Version: 6 TrafficClass: (0) FlowLabel: (0) PayloadLength: 16 NextHeader: 58 HopLimit : XXXXX SourceAddress: **NUT Global IPv6 Address Link2** DestinationAddress: **TN1 Global IPv6 Address with 6to4 Prefix**
ICMPv6 Packet Too Big (length: 1240) Type: 2 Code: 0 Checksum: XXXXX MTU: 1280 **Payload (length: 1232)** Data: the first 1232 bytes of the corresponding IPv6/UDP Packet

Procedure:

0. Set IPv4 MTU on NUT to 1300. As a consequence the tunnel MTU should be 1280.
1. TN1 sends an “IPv6/UDP” Packet to the NUT for TN2Bis with IPv6 Packet size = 1281 bytes.

Observable Results:

- **Step 1:** The NUT MUST send an ICMPv6 Packet too big Message to TN1 with MTU = 1280 and drop the previous packet.

Test Sequence:

Tester	Link1 [IPv6]	RUT	Link2 [IPv4] MTU == 1300	Tester(IPv6)
	IPv6/UDP to TN2Bis (Packet size = 1281)			
TN1	---------->	1		
	ICMPv6 Packet Too Big			
TN1	<----------	Step 1		

Postambule:

A successful run of this test case should have only modified the NDP cache. Thus, to put the NUT in its initial state it has to be cleaned.

After the test execution, the caches entries will be the following:

ARP Cache	NDP Cache
(TR)	TN1

The ARP Cache may be empty, because no stream has been generated from the 6to4 Network to the IPv4 Network. Nevertheless, we have to keep in mind that TR is the default IPv4 route of the NUT. Thus, it will not be problematic to find TR in the ARP cache...

Possible Problem:

A Possible problem can appear in this test case. The ICMPv6 Too Big packet may not match the waited packet. According to **[RFC2463]**, when processing ICMPv6 messages, Implementations MUST observe the following rule: every ICMPv6 error message (type < 128) MUST includes as much of the IPv6 offending (invoking) packet (the packet that caused the error) as will fit without making the error message packet exceed the minimum IPv6 MTU. It means that the first 1232 bytes of the corresponding IPv6/UDP Packet has to be included in the ICMPv6 Too Big packet to fit the minimum IPv6 MTU (1280 Bytes).

5.2.1.3 ICMPv6 Packet Too big (2) **

Purpose:

Set the IPv4 MTU on NUT to 1301. As a consequence the tunnel MTU should be 1281. If the NUT receives an IPv6 Packet greater than 1281 bytes, it MUST send an ICMPv6 Packet too big Message with MTU = 1281 and drop the previous packet. If a static value of 1280 is generated for the tunnel MTU, this test should be skipped.

References:

- **[RFC2893]** ***Section 3.2*****:**

  "**if** (IPv4 path MTU - 20) is less than or equal to 1280

  if packet is larger than 1280 bytes

  Send IPv6 ICMP "packet too big" with MTU = 1280.

  Drop packet.

  else

  Encapsulate but do not set the Don't Fragment flag in the IPv4 header. The resulting IPv4 packet might be fragmented by the IPv4 layer on the encapsulating node or by some router along the IPv4 path.

  endif

  else

  if packet is larger than (IPv4 path MTU - 20)

  Send IPv6 ICMP "packet too big" with

  MTU = (IPv4 path MTU - 20).

  Drop packet.

  else

  Encapsulate and set the Don't Fragment flag in the IPv4 header.

  endif

  endif"

Resource Requirement:

- Packet generator
- Monitor To capture Packets:

Test Requirement:

- If a static value of 1280 is generated for the tunnel MTU, this test should be skipped.
- The IPv4 MTU on NUT has to be set to 1301. As a consequence the tunnel MTU should be 1281. If the NUT is not able to modify the MTU, It may be done using an ICMP Destination Unreachable message sent from TR to NUT as defined in the PATH MTU Discovery specification **[RFC1191].**

Discussion:

IPv6/UDP Packets sent by TN1 to IPv6 Native addresses must be tunneled by the NUT to TN2. The IPv4 MTU on NUT is set to 1301. If the NUT receives an IPv6 Packet from TN1 with at least 1282 bytes, it MUST send an ICMPv6 Packet too big Message with MTU = 1281 and drop the previous packet.

Packets:

- IPv6/UDP (length: 1282 bytes)

IPv6 Header (length: 40)
Version: 6 TrafficClass: (0) FlowLabel: (0)

PayloadLength: 1242 NextHeader: 17 HopLimit : (64) SourceAddress: **TN1 Global IPv6 Address with 6to4 Prefix** DestinationAddress: **TN2Bis Global IPv6 Native Address**
UDP (length: 1242) SourcePort: (1000) DestinationPort: (1000) Length: 1242 Checksum: To calculate **Payload (length: 1234)** Data: 1234 Bytes of data

- ICMPv6 Packet Too Big (length: 1280 bytes)

IPv6 Header (length: 40) Version: 6 TrafficClass: 0 FlowLabel: 0 PayloadLength: 1240 NextHeader: 58 HopLimit : XXXXX SourceAddress: **NUT Global IPv6 Address Link2** DestinationAddress: **TN1 Global IPv6 Address with 6to4 Prefix**
ICMPv6 Packet Too Big (length: 1240) Type: 2 Code: 0 Checksum: XXXXX MTU: 1281 **Payload (length: 1232)** Data: the first 1232 bytes of the corresponding IPv6/UDP Packet

Procedure:

0. Set IPv4 MTU on NUT to 1301
1. TN1 sends an “IPv6/UDP Packet” to the NUT for TN2Bis with IPv6 Packet size = 1282 bytes.

Observable Results:

- **Step 1:** The NUT MUST send an ICMPv6 Packet too big Message to TN1 with MTU = 1281 and drop the previous packet

Test Sequence:

Tester	Link1 [IPv6]	RUT	Link2 [IPv4] MTU == 1301	Tester(IPv6)
	IPv6/UDP to TN2Bis (Packet size = 1282)			
TN1	---------->	1		
	ICMPv6 Packet Too Big			
TN1	<----------	Step 1		

Postambule:

A successful run of this test case should have only modified the NDP cache. Thus, to put the NUT in its initial state it has to be cleaned.

After the test execution, the caches entries will be the following:

ARP Cache	NDP Cache
(TR)	TN1

The ARP Cache may be empty, because no stream has been generated from the 6to4 Network to the IPv4 Network. Nevertheless, we have to keep in mind that TR is the default IPv4 route of the NUT. Thus, it will not be problematic to find TR in the ARP cache...

Possible Problem:

A Possible problem can appear in this test case. The ICMPv6 Too Big packet may not match the waited packet. According to [RFC2463], when processing ICMPv6 messages, Implementations MUST observe the following rule: every ICMPv6 error message (type < 128) MUST includes as much of the IPv6 offending (invoking) packet (the packet that caused the error) as will fit without making the error message packet exceed the minimum IPv6 MTU. It means that the first 1232 bytes of the corresponding IPv6/UDP Packet has to be included in the ICMPv6 Too Big packet to fit the minimum IPv6 MTU (1280 Bytes).

5.2.1.4 ICMPv6 Time Exceeded message

Purpose:

Check that the NUT sends an ICMPv6 Time Exceeded message if it receives a packet with the Hop Limit set to 0 or 1.

References:

- **[RFC2893]** ***Section 3.3***
- **[RFC2463]** ***Section 3.3***

Resource Requirement:

- Packet generator
- Monitor To capture Packets:

Test Requirement:

NONE

Discussion:

ICMPv6 Echo Request sent by TN1 to IPv6 Native addresses must be tunneled by the NUT to TN2 that is the default 6to4 relay. The single-hop model is implemented by having the encapsulating and decapsulating nodes process the IPv6 hop limit field as they would if they were forwarding a packet on to any other datalink. That is, they decrement the hop limit by 1 when forwarding an IPv6 packet. (The originating node and final destination do not decrement the hop limit.). As a consequence, if the Hop Limit is set to 1 or 0 in the Echo Request sent by TN1, an "ICMPv6 Time Exceeded Code 0" must be issued by the NUT to TN1.

Packets:

- ICMPv6 Echo Request (length: 56 bytes)

IPv6 Header (length: 40) Version: 6 TrafficClass: (0) FlowLabel: (0) PayloadLength: 16 NextHeader: 58 HopLimit : 0 or 1 SourceAddress: **TN1 Global IPv6 Address with 6to4 Prefix** DestinationAddress: **TN2Bis Global IPv6 Native Address**
ICMPv6 Echo Request (length: 16) Type: 128 Code: 0 Checksum: To calculate Identifier: (615) SequenceNumber: (0) **Payload (length: 8)** Data: (01234567 89abcdef)

- ICMPv6 Time Exceeded Code 0 (length: 104 bytes)

IPv6 Header (length: 40) Version: 6 TrafficClass: 0 FlowLabel: 0 PayloadLength: 64

NextHeader: 58 HopLimit : XXXXX SourceAddress: **NUT Global IPv6 Address (Link1 or Link2)/NUT Link-local IPv6 address Link1** DestinationAddress: **TN1 Global IPv6 Address with 6to4 Prefix**
ICMPv6 Time Exceeded (length: 64) Type: 3 Code: 0 Checksum: XXXXX Unused: 0 **Payload (length: 56)** Data: the corresponding ICMPv6 Echo Request

Procedure:

1. TN1 sends an “ICMPv6 Echo Request” with Hop Limit set to 0 to the NUT for TN2bis.
2. TN1 sends an “ICMPv6 Echo Request” with Hop Limit set to 1 to the NUT for TN2bis.

Observable Results:

- **Step 1:** The NUT replies with an “ICMPv6 Time Exceeded Code 0” to TN1. The source address should be the **NUT Global IPv6 Address Link1** or the **NUT Link-local IPv6 address Link1.** The offending packet included in the “ICMPv6 Time Exceeded Code 0” should be the one which causes the error (ie the “ICMPv6 Echo Request” with Hop Limit set to 0)
- **Step 2:** The NUT replies with an “ICMPv6 Time Exceeded Code 0” to TN1. The source address should be the **NUT Global IPv6 Address Link1, NUT Global IPv6 Address Link2** or the **NUT Link-local IPv6 address Link1.** The offending packet included in the “ICMPv6 Time Exceeded Code 0” should be the one which causes the error (ie the “ICMPv6 Echo Request” with Hop Limit set to 1 or 0).

Test Sequence:

Tester	Link1 [IPv6]	RUT	Link2 [IPv4]	Tester(IPv6)
	ICMPv6 Echo Request			
TN1	---------->	1		
	ICMPv6 Time Exceeded code 0			
TN1	<----------	Step 1		
	ICMPv6 Echo Request			
TN1	---------->	2		
	ICMPv6 Time Exceeded code 0			
TN1	<----------	Step 2		

Postambule:

A successful run of this test case should have only modified the NDP and ARP caches. Thus, to put the NUT in its initial state they have to be cleaned.

After the test execution, their entry will be the following:

ARP Cache	NDP Cache
(TR)	TN1

The ARP Cache may be empty, because the Hop Limit decrementation will be done before the forwarding procedure. Nevertheless, we have to keep in mind that TR is the default IPv4 route of the NUT. Thus, it will not be problematic to find TR in the ARP cache...

Possible Problem:

Possible problems can appear in this test case. The ICMPv6 Time Exceeded Code 0 packet may not match the awaited packet:

- The ICMPv6 Time Exceeded Code 0 does not contain the full ICMPv6 Echo Request packet. According to **[RFC2463]**, when processing ICMPv6 messages, Implementations MUST observe the following rule: “every ICMPv6 error message (type < 128) MUST includes as much of the IPv6 offending (invoking) packet (the packet that caused the error) as will fit without making the error message packet exceed the minimum IPv6 MTU”. However, with the full ICMPv6 Echo Request packet, the ICMPv6 Time Exceeded Code 0 length is 104 bytes which is less than the minimum IPv6 MTU 1280 bytes.

5.2.1.5 Handling ICMPv4 Destination Unreachable Message (*)

Purpose:

Check that ICMPv4 Destination Unreachable message coming from the v4 Network, whose the packet that caused the error is an IPv6 over IPv4 packet MAY generate an ICMPv6 Destination Unreachable packet. Indeed, if encapsulated part of the ICMPv4 Destination Unreachable Message contains enough data, the encapsulating node MAY extract the encapsulated IPv6 packet and use it to generate an ICMPv6 message directed back to the originating IPv6 node.

References: [RFC2893] ***Section 3.4***

Resource Requirement:

- Packet generator
- Monitor To capture Packets:

Test Requirement:

NONE.

Discussion:

This test is not a strict requirement. If the NUT receives an ICMPv6 Destination Unreachable Packet which include enough data the encapsulating node MAY extract the encapsulated IPv6 packet and use it to generate an IPv6 ICMP message directed back to the originating IPv6 node.

Packets:

- ICMPv6 Echo Request (length: 56 bytes)

IPv6 Header (length: 40) Version: 6 TrafficClass: (0) FlowLabel: (0) PayloadLength: 16 NextHeader: 58 HopLimit : (64) SourceAddress: **TN1 Global IPv6 Address with 6to4 Prefix** - DestinationAddress: **TN Global IPv6 Address with 6to4 Prefix4 (6to4 Prefix4 is unreachable)**
ICMPv6 Echo Request (length: 16) Type: 128 Code: 0 Checksum: To calculate Identifier: (625) SequenceNumber: (0) **Payload (length: 8)** Data: (01234567 89abcdef)

- ICMPv6 Echo Request Tunneled (length: 76 bytes)

IPv4 Header (length: 20) Version: 4 IHL: 5 TypeOfService: 0 TotalLength: 76 Identifier: XXXXX Reserved: 0 DF: 0

MF: 0 FragmentOffset: 0 TTL: XXXXX Protocol: 41 HeaderChecksum: XXXXX SourceAddress: **NUT IPv4 Address Link 2** DestinationAddress: **TN IPv4 Address**
Same as the ICMPv6 Echo Request Packet (length: 56) (Except than the Hop Limit is one less. ie set to 63 in our case)

- ICMPv4 Destination Unreachable (length: 104 bytes)

IPv4 Header (length: 20) Version: 4 IHL: 5 TypeOfService: 0 TotalLength: 104 Identifier: (519) Reserved: 0 DF: 0 MF: 0 FragmentOffset: 0 TTL: (64) Protocol: 41 HeaderChecksum: To calculate SourceAddress: **TR IPv4 Address Link 2** DestinationAddress: **NUT IPv4 Address**
ICMPv4 Destination Unreachable (length: 84) Type: 3 Code: 1 Checksum: To calculate Unused: 0 **Payload (length: 76)** Data: the corresponding ICMPv6 Echo Request tunneled

- ICMPv6 Destination Unreachable (length: 104 bytes)

IPv6 Header (length: 40) Version: 6 TrafficClass: 0 FlowLabel: 0 PayloadLength: 16 NextHeader: 58 HopLimit : XXXXX SourceAddress: **NUT Global IPv6 Address or NUT Link-local IPv6 Address** DestinationAddress: **TN1 Global IPv6 Address with 6to4 Prefix**
ICMPv6 Destination Unreachable (length: 64) Type: 1 Code: 3

Checksum: XXXXX Unused: 0 **Payload (length: 56)** Data: the corresponding ICMPv6 Echo Request

Procedure:

1. TN1 sends an “ICMPv6 Echo Request” to the NUT for one unreachable host (destination address is Global IPv6 Address with unreachable 6to4 Prefix4).
2. TR sends an “ICMPv4 Destination Unreachable” Packet to the NUT.

Observable Results:

- **Step 1:** This previous packet will be tunneled by the NUT in an “ICMPv6 Echo Request tunneled” and transmit to TR
- **Step 2:** The NUT MAY send an “ICMPv6 Destination Unreachable” to TN1.

Test Sequence:

Tester	Link1 [IPv6]	RUT	Link2 [IPv4]	Tester(IPv6)
	ICMPv6 Echo Request			
TN1	---------->	1		
			ICMPv6 Echo Request Tunneled	
TN1		Step 1	---------->	TR
			ICMPv4 Destination Unreachable	
TN1		2	<----------	TR
	ICMPv6 Destination Unreachable			
	<----------	Step 2		

Postambule:

A successful run of this test case should have only modified the NDP and ARP caches. Thus, to put the NUT in its initial state they have to be cleaned.

After the test execution, their entry will be the following:

ARP Cache	NDP Cache
TR	TN1

Possible Problem:

A Possible problem can appear in this test case. The ICMPv6 Destination Unreachable packet may not match the waited packet. Indeed, The ICMPv6 Destination Unreachable packet may not contain the full ICMPv6 Echo Request packet. According to **[RFC2463]**, when processing ICMPv6 messages, Implementations MUST observe the following rule: every ICMPv6 error message (type < 128) MUST includes as much of the IPv6 offending (invoking) packet (the packet that caused the error) as will fit without making the error message packet exceed the minimum IPv6 MTU. However, with the full ICMPv6 Echo Request packet, the length of the ICMPv6 Destination Unreachable packet is 104 bytes, which is less than the minimum IPv6 MTU 1280 bytes.

5.2.2 From IPv6 Network of NUT

5.2.2.1 ICMPv6 Destination Unreachable

Purpose:

Check that the NUT properly generates an ICMPv6 Destination Unreachable message

References:

- **[RFC2463]** ***Section 3.1***

Resource Requirement:

- Packet generator
- Monitor To capture Packets:

Test Requirement:

- Remove the Default IPv6 Route on the NUT.
- The NUT MUST not be listening on port 7777

Discussion:

An ICMPv6 Destination Unreachable Message SHOULD be generated by a router in the originating node in response to a packet that cannot be delivered to a destination for reasons other than congestion.

The ICMPv6 Code may be set to:

- 0 - no route to destination
- 1 - communication with destination administratively prohibited
- 3 - address unreachable
- 4 - port unreachable

Packets:

- ICMPv6 Echo Request Tunneled (length: 1301 bytes)

IPv4 Header (length: 20)
Version: 4 IHL: 5 TypeOfService: 0 TotalLength: 1301 Identifier: (517) Reserved: 0 DF: 0 MF: 0 FragmentOffset: 0 TTL: (64) Protocol: 41 HeaderChecksum: To calculate SourceAddress: **TN3 IPv4 Address** DestinationAddress: **NUT IPv4 Address Link 2**
IPv6 Header (length: 40)
Version: 6 TrafficClass: (0) FlowLabel: (0) PayloadLength: 1241 NextHeader: 58 HopLimit : (64)

SourceAddress: **TN3Bis Global IPv6 Address with 6to4 Prefix3** DestinationAddress: **TN2Bis Global IPv6 Native Address/TN Global IPv6 Address with 6to4 Prefix**
ICMPv6 Echo Request (length: 1241) Type: 128 Code: 0 Checksum: To calculate Identifier: (630) SequenceNumber: (0) **Payload (length: 1233)** Data: 1233 Bytes Of Data

- IPv6/UDP tunneled (length: 76 bytes)

IPv4 Header (length: 20) Version: 4 IHL: 5 TypeOfService: 0 TotalLength: 76 Identifier: (517) Reserved: 0 DF: 0 MF: 0 FragmentOffset: 0 TTL: (64) Protocol: 41 HeaderChecksum: To calculate SourceAddress: **TN3 IPv4 Address** DestinationAddress: **NUT IPv4 Address Link 2**
IPv6 Header (length: 40) Version: 6 TrafficClass: (0) FlowLabel: (0) PayloadLength: 16 NextHeader: 17 HopLimit : (64) SourceAddress: **TN3Bis Global IPv6 Address with 6to4 Prefix3** DestinationAddress: **NUT Global IPv6 Address with 6to4 Prefix (Link1** or **Link2)**
UDP (length: 16) SourcePort: (1000) DestinationPort: **7777** Length: 16 Checksum: To calculate **Payload (length: 8)** Data: (01234567 89abcdef)

- ICMPv6 Destination Unreachable Tunneled (length: 124 bytes)

IPv4 Header (length: 20) Version: 4 IHL: 5 TypeOfService: 0 TotalLength: 124 Identifier: (517) Reserved: 0 DF: 0 MF: 0 FragmentOffset: 0 TTL: (64) Protocol: 41 HeaderChecksum: To calculate SourceAddress: **NUT IPv4 Address Link 2** DestinationAddress: **TN3 IPv4 Address**
IPv6 Header (length: 40) Version: 6 TrafficClass: 0 FlowLabel: 0 PayloadLength: 64 NextHeader: 58 HopLimit : XXXXX SourceAddress: **NUT Global IPv6 Address with 6to4 Prefix (Link1 or Link2)** DestinationAddress: **TN3bis Global IPv6 Address with 6to4 Prefix**
ICMPv6 Destination Unreachable (length: 64) Type: 1 Code: **CODE** Checksum: XXXXX Unused: 0 **Payload (length: 56)** Data: the corresponding offending packet (ICMPv6 Echo Request or IPv6/UDP)

Procedure:

0. Remove the Default IPv6 Route on the NUT.
1. TN3 sends an “ICMPv6 Echo Request Tunneled” packet from TN3Bis to TN2Bis (Native IPv6 Global Address) through the NUT. (ie the MAC address used in the destination field of the Ethernet header is the NUT MAC address LINK2).
2. TN3 sends an “ICMPv6 Echo Request Tunneled” packet from TN3Bis to TN (Unreachable Host located in the 6to4 Network).
3. TN3 sends some “IPv6/UDP Tunneled” Packet from TN3Bis to the NUT (The UDP Port is set to 7777 and the NUT MUST not be listening on this port). The “IPv6/UDP Tunneled” packets use the following addresses:
    i. The Destination Address is **NUT Global IPv6 Address Link1.**
    ii. The Destination Address is **NUT Global IPv6 Address Link2.**

Observable Results:

- **Step 1:** The NUT replies with an “ICMPv6 Destination Unreachable Tunneled” to TN3. The IPv6 source address should be the **NUT Global IPv6 Address Link1 or Link2**. The Code Field (**CODE**) SHOULD be set to 0. The offending packet included in the “ICMPv6 Destination Unreachable Tunneled” should be the one which causes the error (ie the “ICMPv6 Echo Request part included in the “ICMPv6 Echo Request Tunneled” packet).

- **Step 2:** The NUT may send Neighbor solicitation Packet on the 6to4 Network for getting MAC Address of TN. However TN will not respond. The NUT replies with an "ICMPv6 Destination Unreachable Tunneled" to TN3. The source address should be the **NUT Global IPv6 Address Link1 or Link2.** The Code Field (**CODE**) SHOULD be set to 3. The offending packet included in the "ICMPv6 Destination Unreachable Tunneled" should be the one which causes the error (ie the "ICMPv6 Echo Request part included in the "ICMPv6 Echo Request Tunneled" packet).
- **Step 3:** The NUT replies with an "ICMPv6 Destination Unreachable Tunneled" to TN3 to each packet. The IPv6 source addresses should be the the same as the destination addresses of the offending packet. The Code Field (**CODE**) SHOULD be set to 4. The offending packet included in the "ICMPv6 Destination Unreachable" should be the one which causes the error (ie the "IPv6/UDP part included in the "IPv6/UDP Tunneled" packet). The source addresses are:
    i. The **NUT Global IPv6 Address Link1.**
    ii. The **NUT Global IPv6 Address Link2.**

Test Sequence:

Tester	Link1 [IPv6]	RUT	Link2 [IPv4]	Tester(IPv6)
		1	ICMPv6 Echo Request Tunneled <----------	TN3(TN3Bis)
		Step 1	ICMPv6 Destination Unreachable Tunneled ---------->	TN3(TN3Bis)
		2	ICMPv6 Echo Request Tunneled <----------	TN3(TN3Bis)
		Step 2	ICMPv6 Destination Unreachable Tunneled ---------->	TN3(TN3Bis)
		3	IPv6/UDP Tunneled <----------	TN3(TN3Bis)
		Step 3	ICMPv6 Destination Unreachable Tunneled ---------->	TN3(TN3Bis)
		4	IPv6/UDP Tunneled <----------	TN3(TN3Bis)
		Step 4	ICMPv6 Destination Unreachable Tunneled ---------->	TN3(TN3Bis)

Postambule:

A successful run of this test case should have only modified the ARP cache. Thus, to put the NUT in its initial state it has to be cleaned.

After the test execution, the cache entries entry will be the following:

ARP Cache	NDP Cache
TR	

After this test you need to set the previous IPv6 default routes.

Possible Problem:

- Possible problems can appear in this test case. The "ICMPv6 Destination Unreachable Tunneled" packet may not match the awaited packet. According to **[RFC2463]**, when processing ICMPv6 messages, Implementations MUST observe the following rule: "every ICMPv6 error message (type < 128) MUST includes as much of the IPv6 offending (invoking) packet (the packet that caused the error) as will fit without making the error message packet exceed the minimum IPv6 MTU". However, with the full offending packet, (the "ICMPv6 Echo Request part included in the "ICMPv6 Echo Request Tunneled" packet), packet length of "ICMPv6 Destination Unreachable Tunneled" is 124 bytes which is less than the IPv4 MTU 1500 bytes. Moreover, the IPv6 MTU on the 6to4 pseudo-interface is at least 1280 bytes.

5.2.2.2 ICMPv6 Packet Too Big **

Purpose:

Set the IPv4 MTU of Link2 on NUT to 1301 and the IPv6 MTU of Link1 to 1280 (IPv6 Minimum Link MTU). As a consequence the tunnel MTU should be 1281. If the NUT receives an Encapsulated IPv6 Packet greater of 1281 bytes, it MUST send an ICMPv6 Packet too big Message with MTU = 1280 and drop the previous packet. If a static value of 1280 is generated for the tunnel MTU, this test should be skipped.

References:

- **[RFC2463]** ***Section 3.2***

Resource Requirement:

- Packet generator
- Monitor To capture Packets:

Test Requirement:

- If a static value of 1280 is generated for the tunnel MTU, this test should be skipped.
- The IPv4 MTU on NUT has to be set to 1301. As a consequence the tunnel MTU should be 1281. If the NUT is not able to modify the MTU, It may be done using an ICMP Destination Unreachable message sent from TR to NUT as defined in the PATH MTU Discovery specification **[RFC1191].**
- The IPv6 MTU on Link1 of the NUT has to be set to 1280. If the NUT is not able to modify the MTU, It may be done using an ICMP Destination Unreachable message sent from TR to NUT as defined in the PATH MTU Discovery specification **[RFC1981].**

Discussion:

The IPv4 MTU on NUT has to be set to 1301. As a consequence the tunnel MTU should be 1281. If the NUT is not able to modify the IPv4 MTU, It may be done using an ICMP Destination Unreachable message sent from TR to NUT as defined in the PATH MTU Discovery specification [RFC1191]. If a static value of 1280 is generated for the tunnel MTU, this test should be skipped.

The IPv6 MTU on NUT has to be set to 1280. If the NUT is not able to modify the IPv6 MTU, It may be done using an ICMPv6 Destination Unreachable message sent from TN1 to the NUT as defined in the PATH MTU Discovery specification [RFC1981].

Packets:

- ICMPv6 Echo Request Tunneled (length: 1301 bytes)

IPv4 Header (length: 20)
Version: 4 IHL: 5 TypeOfService: 0 TotalLength: 1301 Identifier: (517) Reserved: 0 DF: 0 MF: 0 FragmentOffset: 0 TTL: (64) Protocol: 41 HeaderChecksum: To calculate SourceAddress: **TN2 IPv4 Address** DestinationAddress: **NUT IPv4 Address Link 2**
IPv6 Header (length: 40)
Version: 6 TrafficClass: (0) FlowLabel: (0)

PayloadLength: 1241 NextHeader: 58 HopLimit : (64) SourceAddress: **TN2Bis Global IPv6 Native Address** DestinationAddress: **TN1 Global IPv6 Address with 6to4 Prefix**
ICMPv6 Echo Request (length: 1241) Type: 128 Code: 0 Checksum: To calculate Identifier: (518) SequenceNumber: (0) **Payload (length: 1233)** Data: 1233 Bytes Of Data

- ICMPv6 Packet too big Tunneled (length: 1281 bytes)

IPv4 Header (length: 20) Version: 4 IHL: 5 TypeOfService: 0 TotalLength: 1281 Identifier: XXXXX Reserved: 0 DF: 0 MF: 0 FragmentOffset: 0 TTL: XXXXX Protocol: 41 HeaderChecksum: XXXXX SourceAddress: **NUT IPv4 Address Link 2** DestinationAddress: **TN2 IPv4 Address**
IPv6 Header (length: 40) Version: 6 TrafficClass: 0 FlowLabel: 0 PayloadLength: 1221 NextHeader: 58 HopLimit : XXXXX SourceAddress: **NUT Global IPv6 Address Link1** DestinationAddress: **TN2Bis Global IPv6 Native Address**
ICMPv6 Packet too Big (length: 1221) Type: 2 Code: 0 Checksum: XXXXX MTU: 1280 **Payload (length: 1213)** Data: the first 1213 Bytes of the 1281 of the IPv6 Packet contained in the ICMPv6 Echo Request Tunneled Message

Procedure:

0. Set IPv4 MTU on NUT to 1301 and IPv6 MTU on Link1 to 1280

1. TN2 sends an “ICMPv6 Echo Request tunneled” from TN2Bis to TN1 (Packet size is 1301 bytes with 1281 bytes for the IPv6 Packet).

Observable Results:

- **Step 1:** The NUT MUST send an “ICMPv6 Packet too big Tunneled” Message to TN2 with MTU = 1280 and drop the previous packet.

Test Sequence:

Tester	Link1 [IPv6] MTU == 1280	RUT	Link2 [IPv4] MTU == 1301	Tester(IPv6)
			ICMPv6 Echo Request Tunneled	
		1	<----------	TN2(TN2Bis)
			ICMPv6 Packet Too Big Tunneled	
		Step 1	---------->	TN2(TN2Bis)

Postambule:

A successful run of this test case should have only modified the NDP and ARP caches. Thus, to put the NUT in its initial state they have to be cleaned.

After the test execution, their entry will be the following:

ARP Cache	NDP Cache
TR	(TN1)

The NDP Cache may be empty, because the packet has been discarded.

Possible Problem:

Possible problems can appear in this test case. The “ICMPv6 Packet too big Tunneled” packet may not match the awaited packet. According to **[RFC2463]**, when processing ICMPv6 messages, Implementations MUST observe the following rule: every ICMPv6 error message (type < 128) MUST include as much of the IPv6 offending (invoking) packet (the packet that caused the error) as will fit without making the error message packet exceed the minimum IPv6 MTU. As a consequence, in this test the “ICMPv6 Packet too big Tunneled” SHOULD contain the first 1213 Bytes of the IPv6 Packet included in the ICMPv6 Echo Request Tunneled Message.

5.2.2.3 ICMPv6 Time Exceeded message

Purpose:

Check that the NUT sends an ICMPv6 Time Exceeded message if it receives an IPv6 tunneled packet with the Hop Limit set to 0 or 1.

References:

- **[RFC2893]** ***Section 3.3***
- **[RFC2463]** ***Section 3.3***

Resource Requirement:

- Packet generator
- Monitor To capture Packets:

Test Requirement:

NONE

Discussion:

ICMPv6 Echo Request sent by IPv6 Native addresses to TN1 or to the NUT must be tunneled by the nearest 6to4 relay of this host (TN2). The single-hop model is implemented by having the encapsulating and decapsulating nodes process the IPv6 hop limit field as they would if they were forwarding a packet on to any other datalink. That is, they decrement the hop limit by 1 when forwarding an IPv6 packet. Thus, if the Hop Limit is set to 1 or 0 in the Echo Request sent by TN2bis, an "ICMPv6 Time Exceeded Code 0" must be tunneled by the NUT to TN2bis.

Comment: As describe in **[6to4Security]**, This functionnality permit to anyone to generate attacks such as:

- Unidirectional source address spoofing
- Denial-of-service attack reflection against native IPv6 nodes.

Indeed, the 6to4 router cannot distinguish 6to4 Relay routers from malicious nodes sending IPv4-encapsulated IPv6 traffic directly to the router.

Packets:

- ICMPv6 Echo Request Tunneled (length: 76 bytes)

IPv4 Header (length: 20)
Version: 4 IHL: 5 TypeOfService: 0 TotalLength: 76 Identifier: (517) Reserved: 0 DF: 0 MF: 0 FragmentOffset: 0 TTL: (64) Protocol: 41 HeaderChecksum: To calculate SourceAddress: **TN2 IPv4 Address** DestinationAddress: **NUT IPv4 Address Link 2**
IPv6 Header (length: 40)
Version: 6 TrafficClass: (0) FlowLabel: (0) PayloadLength: 16 NextHeader: 58 HopLimit : 1 or 2

SourceAddress: **TN2Bis Global IPv6 Native Address** DestinationAddress: **TN1 Global IPv6 Address with 6to4 Prefix**
ICMPv6 Echo Request (length: 16) Type: 128 Code: 0 Checksum: To calculate Identifier: (633) SequenceNumber: (0) **Payload (length: 8)** Data: (01234567 89abcdef)

- ICMPv6 Time Exceeded Code 0 Tunneled (length: 124 bytes)

IPv4 Header (length: 20) Version: 4 IHL: 5 TypeOfService: 0 TotalLength: 76 Identifier: XXXXX Reserved: 0 DF: 0 MF: 0 FragmentOffset: 0 TTL: XXXXX Protocol: 41 HeaderChecksum: XXXXX SourceAddress: **NUT IPv4 Address Link 2** DestinationAddress: **TN2 IPv4 Address**
IPv6 Header (length: 40) Version: 6 TrafficClass: 0 FlowLabel: 0 PayloadLength: 16 NextHeader: 58 HopLimit : XXXXX SourceAddress: **NUT Global IPv6 Address (Link1 or Link2)** DestinationAddress: **TN2Bis Global IPv6 Native Address**
ICMPv6 Time Exceeded (length: 64) Type: 3 Code: 0 Checksum: XXXXX Unused: 0 **Payload (length: 56)** Data: the corresponding ICMPv6 Echo Request with the Hop Limit field set to the value it receives by TR.

Procedure:

1. TN2 sends an “ICMPv6 Echo Request tunneled” from TN2Bis with Hop Limit set to 1 to the NUT for TN1. (TR will decrement this value to 0 before to forward it to the NUT)
2. TN2 sends an “ICMPv6 Echo Request tunneled” from TN2Bis with Hop Limit set to 2 to the NUT for TN1. (TR will decrement this value to 1 before to forward it to the NUT)

Observable Results:

- **Step 1:** The NUT replies with an “ICMPv6 Time Exceeded Code 0 tunneled” to TN2. The source address should be the **NUT Global IPv6 Address Link2.** The offending packet included in the “ICMPv6 Time Exceeded Code 0” should be the one which causes the error (ie the “ICMPv6 Echo Request” with Hop Limit set to 0)
- **Step 2:** The NUT replies with an “ICMPv6 Time Exceeded Code 0 tunneled” to TN2. The source address should be the **NUT Global IPv6 Address Link1** or the **NUT Global IPv6 Address Link2.** The offending packet included in the “ICMPv6 Time Exceeded Code 0” should be the one which causes the error (ie the “ICMPv6 Echo Request” with Hop Limit set to 1 or 0).

Test Sequence:

Tester	Link1 [IPv6]	RUT	Link2 [IPv4]	Tester(IPv6)
			ICMPv6 Echo Request Tunneled	
		1	<----------	TN2(TN2Bis)
			ICMPv6 Time Exceeded Code 0 Tunneled	
		Step 1	---------->	TN2(TN2Bis)
			ICMPv6 Echo Request Tunneled	
		2	<----------	TN2(TN2Bis)
			ICMPv6 Time Exceeded Code 0 Tunneled	
		Step 2	---------->	TN2(TN2Bis)

Postambule:

A successful run of this test case should have only modified the NDP and ARP caches. Thus, to put the NUT in its initial state they have to be cleaned.

After the test execution, their entry will be the following:

ARP Cache	NDP Cache
TR	(TN1)

The NDP Cache may be empty, because the Hop Limit decrementation will be done before the forwarding procedure.

Possible Problem:

Possible problems can appear in this test case. The ICMPv6 Time Exceeded Code 0 tunneled packet may not match the waited packet.

- The ICMPv6 Time Exceeded Code 0 tunneled packet does not contain the full ICMPv6 Echo Request packet. According to **[RFC2463]**, when processing ICMPv6 messages, Implementations MUST observe the following rule: every ICMPv6 error message (type < 128) MUST includes as much of the IPv6 offending (invoking) packet (the packet that caused the error) as will fit without making the error message packet exceed the minimum IPv6 MTU. However, with the full ICMPv6 Echo Request packet, the length of ICMPv6 Time Exceeded Code 0 part is 104 bytes which is less than the minimum IPv6 MTU 1280 bytes.

5.3 Addressing

Although, this test suite does not take into account **[6to4Security],** it remains possible to test security aspects described in **[RFC3056]**. In particular, any 6to4 traffic whose source or destination address embeds a V4ADDR which is not in the format of a global unicast address MUST be silently discarded by both encapsulators and decapsulators. Specifically, this means that IPv4 addresses defined in **[RFC1918]**, broadcast, subnet broadcast, multicast and loopback addresses are unacceptable.

This section is used to test the 6to4 addressing mechanism as well as the filtering rules that the NUT should respect.

5.3.1 From IPv6 Network of NUT

5.3.1.1 Outgress Filtering from source

Purpose:

Check that 6to4 traffic coming from the 6to4 network, whose source address embeds a **V4ADDR**, which is not in the format of a global unicast address is silently discarded by the NUT.

References:

- **[RFC3056]** ***Page 19***

  "In any case, any 6to4 traffic whose source or destination address embeds a V4ADDR, which is not in the format of a global unicast address, MUST be silently discarded by both encapsulators and decapsulators. Specifically, this means that IPv4 addresses defined in [RFC1918], broadcast, subnet broadcast, multicast and loopback addresses are unacceptable."

Resource Requirement:

- Packet generator
- Monitor To capture Packets:

Test Requirement:

NONE

Discussion:

Packets send by TN1 whose source address embeds a **V4ADDR**, which is not in the format of a global unicast address MUST be silently discarded by the encapsulator. It means Undefined Address, Loopback Address, Multicast Address All hosts, Multicast Address All routers, Broadcast Address. This check avoids the possibly use of the NUT to broadcast packets on the Internet v4.

Packets:

- ICMPv6 Echo Request (length: 56 bytes)

IPv6 Header (length: 40)
Version: 6 TrafficClass: (0) FlowLabel: (0) PayloadLength: 16 NextHeader: 58 HopLimit : (64) SourceAddress: **IPv6_ SOURCE_ADDR** DestinationAddress: **TN2Bis Global IPv6 Native Address**
ICMPv6 Echo Request (length: 16)
Type: 128 Code: 0 Checksum: To calculate Identifier: (712)

SequenceNumber: (0) **Payload (length: 8)** Data: (01234567 89abcdef)

Procedure:

The test procedure is the following:

1. TN1 sends an “ICMPv6 Echo Request” with source address **IPv6_SOURCE_ADDR** to TN2Bis.

Repeat **Step 1** using the following values for **IPv6_ SOURCE_ADDR:**

- **2002::1 (IPv4 Address is 0.0.0.0 - Undefined Address)**
- **2002:7f00:0001::1 (IPv4 Address is 127.0.0.1 - Loopback Address)**
- **2002:e000:0001::1 (IPv4 Address is 224.0.0.1 - Multicast Address All hosts)**
- **2002:e000:0002::1 (IPv4 Address is 224.0.0.2 - Multicast Address All routers)**
- **2002:83FE:C8FF::1 (IPv4 Address is 131.254.200.255 - Broadcast Address)**
- **2002:0a00:0001::1 (IPv4 Address is 10.0.0.1 - Private Address)**
- **2002:AC10:0001::1 (IPv4 Address is 172.16.0.1 - Private Address)**
- **2002:C0A8:0001::1 (IPv4 Address is 192.168.0.1 - Private Address)**
- **2002:A9FE:0001::1 (IPv4 Address is 169.254.0.1 – DHCP Link-local)**

Observable Results:

- **Step 1:** The NUT MUST not forward these Packets and MUST silently discard each packet.

Test Sequence:

Tester	Link1 [IPv6]	RUT	Link2 [IPv4]	Tester(IPv6)
TN1	ICMPv6 Echo Request ---------->	N Times $N \in \mathbb{N}$ $N \geq 0$		

Postambule:

A successful run of this test case should have only modified the NDP and ARP caches. Thus, to put the NUT in its initial state they have to be cleaned.

After the test execution, their entry will be the following:

ARP Cache	NDP Cache
(TR)	TN1

The ARP Cache may be empty, because no stream has been generated from the 6to4 Network to the IPv4 Network. Nevertheless, we have to keep in mind that TR is the default IPv4 route of the NUT. Thus, it will not be problematic to find TR in the ARP cache...

5.3.1.2 Outgress Filtering from destination

Purpose:

Check that 6to4 traffic coming from the 6to4 network, whose destination address embeds a **V4ADDR**, which is not in the format of a global unicast address is silently discarded by the NUT.

References:

- **[RFC3056]** ***Page 19***

  "In any case, any 6to4 traffic whose source or destination address embeds a V4ADDR, which is not in the format of a global unicast address, MUST be silently discarded by both encapsulators and decapsulators. Specifically, this means that IPv4 addresses defined in [RFC1918], broadcast, subnet broadcast, multicast and loopback addresses are unacceptable."

Resource Requirement:

- Packet generator
- Monitor To capture Packets:

Test Requirement:

NONE

Discussion:

Packets send by TN1 whose destination address embeds a **V4ADDR**, which is not in the format of a global unicast address MUST be silently discarded by the encapsulator. It means Undefined Address, Loopback Address, Multicast Address All hosts, Multicast Address All routers, Broadcast Address. This check avoids the possibly use of the NUT to broadcast packets on the Internet v4.

Packets:

- ICMPv6 Echo Request (length: 56 bytes)

IPv6 Header (length: 40)
Version: 6 TrafficClass: (0) FlowLabel: (0) PayloadLength: 16 NextHeader: 58 HopLimit : (64) SourceAddress: **TN1 Global IPv6 Address with 6to4 Prefix** DestinationAddress: **IPv6_ DESTINATION_ADDR**
ICMPv6 Echo Request (length: 16)
Type: 128 Code: 0 Checksum: To calculate Identifier: (721) SequenceNumber: (0) **Payload (length: 8)** Data: (01234567 89abcdef)

Procedure:

The test procedure is the following:

1. TN1 sends an "ICMPv6 Echo Request" with destination address **IPv6_DESTINATION_ADDR** to the NUT.

Repeat **Step 1** using the following values for **IPv6_ DESTINATION_ADDR:**

- **2002::1 (IPv4 Address is 0.0.0.0 - Undefined Address)**

- **2002:7f00:0001::1 (IPv4 Address is 127.0.0.1 - Loopback Address)**
- **2002:e000:0001::1 (IPv4 Address is 224.0.0.1 - Multicast Address All hosts)**
- **2002:e000:0002::1 (IPv4 Address is 224.0.0.2 - Multicast Address All routers)**
- **2002:83FE:C8FF::1 (IPv4 Address is 131.254.200.255 - Broadcast Address)**
- **2002:0a00:0001::1 (IPv4 Address is 10.0.0.1 - Private Address)**
- **2002:AC10:0001::1 (IPv4 Address is 172.16.0.1 - Private Address)**
- **2002:C0A8:0001::1 (IPv4 Address is 192.168.0.1 - Private Address)**
- **2002:A9FE:0001::1 (IPv4 Address is 169.254.0.1 – DHCP Link-local)**

Observable Results:

- **Step 1:** The NUT MUST not forward these Packets and MUST silently discard each packet.

Test Sequence:

Tester	Link1 [IPv6]	RUT	Link2 [IPv4]	Tester(IPv6)
TN1	ICMPv6 Echo Request ---------->	**N Times** **N**$\in$ N **N**≥ 0		

Postambule:

A successful run of this test case should have only modified the NDP and ARP caches. Thus, to put the NUT in its initial state they have to be cleaned.

After the test execution, their entry will be the following:

ARP Cache	NDP Cache
(TR)	TN1

The ARP Cache may be empty, because no stream has been generated from the 6to4 Network to the IPv4 Network. Nevertheless, we have to keep in mind that TR is the default IPv4 route of the NUT. Thus, it will not be problematic to find TR in the ARP cache...

5.3.2 To IPv6 Network of NUT

5.3.2.1 Ingress Filtering from source

Purpose:

Check that 6to4 traffic whose source address embeds a V4ADDR, which is not in the format of a global unicast address is silently discarded by the decapsulator

References:

- **[RFC3056]** ***Section 9***

Resource Requirement:

- Packet generator
- Monitor To capture Packets:

Test Requirement:

NONE

Discussion:

[RFC3056] requiere that 6to4 traffic whose destination address embeds a V4ADDR which is not in the format of a global unicast address is silently discarded by decapsulators. No more security considerations are defined in [RFC3056]. Nevertheless, it gives the opportunity to add address based packet filtering. In particular **[6to4Security]** gives advices on addresses not to use:

6to4 Address in which the encapsulating IPv4 Adress is not consistent with the encapsulated 2002:: address

- *IPv6 Address of hosts located in the 6to4 network.*
- *IPv6 Loopback, Undefined, Multicast site, link-local, solicited-node address, IPv4 Mapped and Compatible Address: This check will avoid the possible threats to Neighbor Discovery listed in* ***[NDThreats].***

Packets:

- ICMPv6 Echo Request Tunneled (length: 76 bytes)

IPv4 Header (length: 20)

Version: 4
IHL: 5
TypeOfService: 0
TotalLength: 76
Identifier: (522)
Reserved: 0
DF: 0
MF: 0
FragmentOffset: 0
TTL: (64)
Protocol: 41
HeaderChecksum: To calculate
SourceAddress: **TN2 IPv4 Address**
DestinationAddress: **NUT IPv4 Address Link 2**

IPv6 Header (length: 40)

Version: 6
TrafficClass: (0)
FlowLabel: (0)
PayloadLength: 16
NextHeader: 58
HopLimit : (64)
SourceAddress: **IPv6_SOURCE_ADDR**

DestinationAddress: **TN1 Global IPv6 Address with 6to4 Prefix**
ICMPv6 Echo Request (length: 16) Type: 128 Code: 0 Checksum: To calculate Identifier: (730) SequenceNumber: (0) **Payload (length: 8)** Data: (01234567 89abcdef)

Procedure:

The test procedure is the following:

1. TN2 sends an "ICMPv6 Echo Request tunneled" with IPv6 source address **IPv6_SOURCE_ADDR** to the NUT for TN1.

Repeat **Step 1** using the following values for **IPv6_SOURCE_ADDR:**

- *6to4 Address in which V4ADDR is not a global unicast IPv4 address:*
  - **2002::1 (IPv4 Address is 0.0.0.0 - Undefined Address)**
  - **2002:7f00:0001::1 (IPv4 Address is 127.0.0.1 - Loopback Address)**
  - **2002:e000:0001::1 (IPv4 Address is 224.0.0.1 - Multicast Address All hosts)**
  - **2002:e000:0002::1 (IPv4 Address is 224.0.0.2 - Multicast Address All routers)**
  - **2002:83FE:C8FF::1 (IPv4 Address is 131.254.200.255 - Broadcast Address)**
  - **2002:0a00:0001::1 (IPv4 Address is 10.0.0.1 - Private Address)**
  - **2002:AC10:0001::1 (IPv4 Address is 172.16.0.1 - Private Address)**
  - **2002:C0A8:0001::1 (IPv4 Address is 192.168.0.1 - Private Address)**
  - **2002:A9FE:0001::1 (IPv4 Address is 169.254.0.1 – DHCP Link-local)**
- *6to4 Address in which the encapsulating IPv4 Adress is not consistent with the encapsulated 2002:: address:*
  - **2002:83FE:CA01::1 (6to4 Address of one possible host located on TN3 network)**
- *IPv6 Address of hosts located in the 6to4 network:*
  - **TN Global IPv6 Address with 6to4 Prefix (TN is located on the NUT Network)**
  - **NUT Global IPv6 Address Link1 with 6to4 Prefix**
  - **NUT Global IPv6 Address Link2 with 6to4 Prefix**
- *IPv6 Loopback, Undefined, Multicast site, link-local, solicited-node address, IPv4 Mapped and Compatible Address:*
  - **::1 (IPv6 Loopback Adress)**
  - **:: (IPv6 Undefined Address)**
  - **::FFFF:83FE:CA01 (IPv4 mapped Address of TN3)**
  - **::83FE:CA01 (IPv4 Compatible Address of TN3)**
  - **Fe80::200:FF:FE00:a3a3 (Link-Local Address of TN3Bis)**
  - **FF02::1 (Multicast Address All hosts on the link)**
  - **FF02::2 (Multicast Address All routers on the link)**
  - **FF02::9 (Multicast Address All Rinpg routers on the link)**
  - **FEC0::1 (Multicast Address All hosts on the site)**
  - **FEC0::2 (Multicast Address All routers on the site)**
  - **FEC0::9 (Multicast Address All Rinpg routers on the site)**

- **FF02::1:FF00:0000:a3a3 (Multicast Address Solicited Node of TN3Bis)**

Test Sequence:

Tester	Link1 [IPv6]	RUT	Link2 [IPv4]	Tester(IPv6)
		N Times **N**$\in$ N **N**≥ 0	ICMPv6 Echo Request <----------	TN2**(IPv6_SOURCE_ADDR)**

Observable Results:

- **Step 1:** The NUT MUST not forward these Packets and MUST silently discard each one.

Postambule:

A successful run of this test case should have only modified the ARP cache. Thus, to put the NUT in its initial state they have to be cleaned.

After the test execution, the caches entries will be the following:

ARP Cache	NDP Cache
TR	

The NDP Cache MUST be empty, because all packets have to be silently discarded.

5.3.2.2 Ingress Filtering from destination

Purpose:

Check that 6to4 traffic whose destination address embeds a V4ADDR, which is not in the format of a global unicast address is silently discarded by the decapsulator

References:

- [RFC3056] *Section 9*

Resource Requirement:

- Packet generator
- Monitor To capture Packets:

Test Requirement:

NONE

Discussion:

[RFC3056] requiere that 6to4 traffic whose destination address embeds a V4ADDR which is not in the format of a global unicast address is silently discarded by decapsulators. No more security considerations are defined in [RFC3056]. Nevertheless, it gives the opportunity to add address based packet filtering. In particular **[6to4Security]** gives advices on addresses not to use:

- *6to4 Address from an other 6to4 Network.*
- *IPv6 Loopback, Undefined, Multicast site, link-local, solicited-node address, IPv4 Mapped and Compatible Address**:** This check will avoid the possible threats to Neighbor Discovery listed in **[NDThreats].***

Packets:

- ICMPv6 Echo Request Tunneled (length: 76 bytes)

IPv4 Header (length: 20)
Version: 4 IHL: 5 TypeOfService: 0 TotalLength: 76 Identifier: (523) Reserved: 0 DF: 0 MF: 0 FragmentOffset: 0 TTL: (64) Protocol: 41 HeaderChecksum: To calculate SourceAddress: **TN2 IPv4 Address** DestinationAddress: **NUT IPv4 Address Link 2**
IPv6 Header (length: 40)
Version: 6 TrafficClass: (0) FlowLabel: (0) PayloadLength: 16 NextHeader: 58 HopLimit : (64) SourceAddress: **TN2Bis Global IPv6 Native Address** DestinationAddress: **IPv6_DESTINATION_ADDR**
ICMPv6 Echo Request (length: 16)

Type: 128
Code: 0
Checksum: To calculate
Identifier: (754)
SequenceNumber: (0)

Payload (length: 8)
Data: (01234567 89abcdef)

Procedure:

The test procedure is the following:

1. TN2 sends an "ICMPv6 Echo Request tunneled" with IPv6 destination address **IPv6_DESTINATION_ADDR** to the NUT for TN1.

Repeat **Step 1** using the following values for **IPv6_DESTINATION_ADDR:**

- *6to4 Address in which V4ADDR is not a global unicast IPv4 address*
  - **2002::1 (IPv4 Address is 0.0.0.0 - Undefined Address)**
  - **2002:7f00:0001::1 (IPv4 Address is 127.0.0.1 - Loopback Address)**
  - **2002:e000:0001::1 (IPv4 Address is 224.0.0.1 - Multicast Address All hosts)**
  - **2002:e000:0002::1 (IPv4 Address is 224.0.0.2 - Multicast Address All routers)**
  - **2002:83FE:C8FF::1 (IPv4 Address is 131.254.255.255 - Broadcast Address)**
  - **2002:0a00:0001::1 (IPv4 Address is 10.0.0.1 - Private Address)**
  - **2002:AC10:0001::1 (IPv4 Address is 172.16.0.1 - Private Address)**
  - **2002:C0A8:0001::1 (IPv4 Address is 192.168.0.1 - Private Address)**
  - **2002:A9FE:0001::1 (IPv4 Address is 169.254.0.1 – DHCP Link-local)**
- *6to4 Address from an other 6to4 Network*
  - **2002:83FE:CA01::1 (6to4 Address of one possible host located on TN3 network)**
- *IPv6 Loopback, Undefined, Multicast site, link-local, solicited-node address, IPv4 Mapped and Compatible Address:*
  - **::1 (IPv6 Loopback Adress)**
  - **:: (IPv6 Undefined Address)**
  - **::FFFF:83FE:CA01 (IPv4 mapped Address of TN3)**
  - **::83FE:CA01 (IPv4 Compatible Address of TN3)**
  - **Fe80::200:FF:FE00:a3a3 (Link-Local Address of TN3Bis)**
  - **FF02::1 (Multicast Address All hosts on the link)**
  - **FF02::2 (Multicast Address All routers on the link)**
  - **FF02::9 (Multicast Address All Rinpg routers on the link)**
  - **FEC0::1 (Multicast Address All hosts on the site)**
  - **FEC0::2 (Multicast Address All routers on the site)**
  - **FEC0::9 (Multicast Address All Rinpg routers on the site)**
  - **FF02::1:FF00:0000:a3a3 (Multicast Address Solicited Node of TN3Bis)**

Observable Results:

- **Step 1:** The NUT MUST not forward this Packet and MUST silently discard this.

Test Sequence:

Tester	Link1 [IPv6]	RUT	Link2 [IPv4]	Tester(IPv6)
		N Times N∈ N N≥ 0	ICMPv6 Echo Request <----------	TN2(TN2Bis)

Postambule:

A successful run of this test case should have only modified the ARP cache. Thus, to put the NUT in its initial state it has to be cleaned.

After the test execution, the caches entries will be the following:

ARP Cache	NDP Cache
TR	

5.4 Fragmentation Handling & MTU

This part wants to test the Fragmentation and MTU handling of the NUT. It will be done essentialy using **[RFC2893]** ***Section 3.2, Page 11,12:***

"**if** (IPv4 path MTU - 20) is less than or equal to 1280

if packet is larger than 1280 bytes

Send IPv6 ICMP "packet too big" with MTU = 1280.

Drop packet.

else

Encapsulate but do not set the Don't Fragment flag in the IPv4 header. The resulting IPv4 packet might be fragmented by the IPv4 layer on the encapsulating node or by some router along the IPv4 path.

endif

else

if packet is larger than (IPv4 path MTU - 20)

Send IPv6 ICMP "packet too big" with

MTU = (IPv4 path MTU - 20).

Drop packet.

else

Encapsulate and set the Don't Fragment flag in the IPv4 header.

endif

endif"

In this part, it is considered that IPv4 PMTU - 20 = Tunnel MTU. According to **[RFC 2893]** the IPv6 layer in the encapsulating node can view a tunnel as a link layer with an MTU equal to the IPv4 path MTU, minus the size of the encapsulating IPv4 header. So, if a node employs the static value 1280 for pseudo-Interface, it should skip test 5.4.1.4

Tests describing ICMPv6 Error generation (i.e when Too Big IPv6 Packet are sent from The Network of the NUT) from the NUT are already defined in section 5.2 (They are 5.2.1 and 5.2.1.3).

5.4.1 From IPv6 Network of NUT

5.4.1.1 Unfragmentation at IPv4 Level (*)

Purpose:

Check that IPv4 Fragmented Packets are reassembled by the NUT before to be forwarded on the 6to4 Network.

References:

- **[RFC2893]** ***Section 3.6***

Resource Requirement:

- Packet generator
- Monitor To capture Packets:

Test Requirement:

NONE

Discussion:

The NUT is the end-point of automatic tunnel in a 6to4 stream coming from TN2 to TN1. It involves that IPv4 Fragmented Packets are reassembled before to be forwarded on the 6to4 Network.

Packets:

- Fragmented IPv4/IPv6/UDP Packet (1st Fragment) (length: 76 bytes)

IPv4 Header (length: 20) Version: 4 IHL: 5 TypeOfService: 0 TotalLength: 76 Identifier: (528) Reserved: 0 DF: 1 MF: 1 FragmentOffset: 0 TTL: (64) Protocol: 41 HeaderChecksum: To calculate SourceAddress: **TN2 IPv4 Address** DestinationAddress: **NUT IPv4 Address Link 2**
IPv6 Header (length: 40) Version: 6 TrafficClass: (0) FlowLabel: (0) PayloadLength: 16 NextHeader: 17 HopLimit : (64) SourceAddress: **TN2Bis Global IPv6 Native Address** DestinationAddress: **TN1 Global IPv6 Address with 6to4 Prefix**
UDP (length: 16) SourcePort: (1000) DestinationPort: (1000) Length: 16 Checksum: To calculate **Payload (length: 8)** Data: (`01234567 89abcdef`)

- Fragmented IPv4/IPv6/UDP Packet (2nd Fragment) (length: 44 bytes)

IPv4 Header (length: 20) Version: 4 IHL: 5 TypeOfService: 0 TotalLength: 44 Identifier: (528 Same as the previous Fragmented Packet) Reserved: 0 DF: 1 MF: 1 FragmentOffset: 7 TTL: (64) Protocol: 41 HeaderChecksum: To calculate SourceAddress: **TN2 IPv4 Address** DestinationAddress: **NUT IPv4 Address Link 2**

Payload (length: 24) Data: 24 Bytes of data

- Fragmented IPv4/IPv6/UDP Packet (Last Fragment) (length: 44 bytes)

IPv4 Header (length: 20) Version: 4 IHL: 5 TypeOfService: 0 TotalLength: 44 Identifier: (528) Reserved: 0 DF: 1 MF: 0 FragmentOffset: 10 TTL: (64) Protocol: 41 HeaderChecksum: To calculate SourceAddress: **TN2 IPv4 Address** DestinationAddress: **NUT IPv4 Address Link 2**
Payload (length: 8) Data: 8 Bytes of data

- IPv6/UDP (length: 102 bytes)

IPv6 Header (length: 40) Version: 6 TrafficClass: (0) FlowLabel: (0) PayloadLength: 48 NextHeader: 17 HopLimit : (63) **(The Hop Limit is one less than in the corresponding IPv6/UDP Fragmented packet)** SourceAddress: **TN2Bis Global IPv6 Native Address** DestinationAddress: **TN1 Global IPv6 Address with 6to4 Prefix**
UDP (length: 48) SourcePort: (1000) DestinationPort: (1000) Length: 48 Checksum: XXXXX **Payload (length: 40)** Data: The contains of payload of all fragmented IPv4/IPv6 Fragmented Packets

Procedure:

1. TN2 sends to the NUT the first fragment of one IPv4/IPv6/UDP Packet.
2. TN2 sends to the NUT the second and the last fragment of one IPv4/IPv6/UDP Packet.

Observable Results:

- **Step 2:** The NUT waits for receiving all fragments, reassembles all the fragments, desencapsulates this resulting packet and forwards this “IPv6/UDP Packet” to TN1 .

Test Sequence:

Tester	Link1 [IPv6]	RUT	Link2 [IPv4]	Tester(IPv6)
		1	IPv4/IPv6/UDP Fragment 1 <----------	TN2(TN2Bis)
		2	IPv4/IPv6/UDP Fragment 2 <----------	TN2(TN2Bis)
TN1	IPv4/IPv6/UDP <----------	Step 2		

Postambule:

A successful run of this test case should have only modified the NDP and ARP caches. Thus, to put the NUT in its initial state they have to be cleaned.

After the test execution, the caches entries will be the following:

ARP Cache	NDP Cache
TR	TN1

5.4.1.2 Fragmentation at IPv4 Level

Purpose:

Set the IPv4 MTU on NUT to 1299. The Minimum MTU for IPv6 is 1280. If the NUT receives an IPv6 Packet equal to 1280, it MUST fragment it before to forward it on the IPv4 Network.

References:

- **[RFC3056]** ***Section 4***
- **[RFC2893]** ***Section 3.2:***

  "**if** (IPv4 path MTU - 20) is less than or equal to 1280

  if packet is larger than 1280 bytes

  Send IPv6 ICMP "packet too big" with MTU = 1280.

  Drop packet.

  else

  Encapsulate but do not set the Don't Fragment flag in the IPv4 header. The resulting IPv4 packet might be fragmented by the IPv4 layer on the encapsulating node or by some router along the IPv4 path.

  endif

  else

  if packet is larger than (IPv4 path MTU - 20)

  Send IPv6 ICMP "packet too big" with

  MTU = (IPv4 path MTU - 20).

  Drop packet.

  else

  Encapsulate and set the Don't Fragment flag in the IPv4 header.

  endif

  endif"

Resource Requirement:

- Packet generator
- Monitor To capture Packets:

Test Requirement:

- The IPv4 MTU on NUT has to be set to 1299. If the NUT is not able to modify the MTU, It may be done using an ICMP Destination Unreachable message sent from TR to NUT as defined in the PATH MTU Discovery specification [RFC1191].

Discussion:

IPv6/UDP Packets sent by TN1 to IPv6 Native addresses must be tunneled by the NUT to TN2. MTU size considerations are as described for tunnels in **[RFC2893]**. If the IPv6 MTU size proves to be too large for some intermediate IPv4 subnet, IPv4 fragmentation will ensue.

The IPv4 MTU on NUT is set to 1299. Because the Minimum MTU for IPv6 is 1280, if the NUT receives an IPv6 Packet equal to 1280, it MUST fragment it before to forward it on the IPv4 Network.

Packets:

- IPv6/UDP (length: 1280 bytes)

IPv6 Header (length: 40)
Version: 6 TrafficClass: (0)

FlowLabel: (0) PayloadLength: 1240 NextHeader: 17 HopLimit : (64) SourceAddress: **TN1 Global IPv6 Address with 6to4 Prefix** DestinationAddress: **TN2Bis Global IPv6 Native Address**
UDP (length: 1240) SourcePort: (1000) DestinationPort: (1000) Length: 1240 Checksum: To calculate **Payload (length: 1232)** Data: 1232 Bytes of data

- IPv6/UDP (1st Fragment) Tunneled (length: 1292 bytes)

IPv4 Header (length: 20) Version: 4 IHL: 5 TypeOfService: 0 TotalLength: 1292 Identifier: XXXXX Reserved: 0 DF: 0 MF: 1 FragmentOffset: 0 TTL: XXXXX Protocol: 41 HeaderChecksum: XXXXX SourceAddress: **NUT IPv4 Address Link 2** DestinationAddress: **TN2 IPv4 Address**
Payload (length: 1272) Data: The First 1272 bytes of the IPv6/UDP Packet **(Except than the Hop Limit is one less. ie set to 63 in our case)**

- IPv6/UDP (Last Fragment) Tunneled (length: 28 bytes)

IPv4 Header (length: 20) Version: 4 IHL: 5 TypeOfService: 0 TotalLength: 28 Identifier: XXXXX (Same as IPv6/UDP First Fragment) Reserved: 0 DF: 0 MF: 0 FragmentOffset: 159 (in 8-octet units) TTL: XXXXX Protocol: 41 HeaderChecksum: XXXXX SourceAddress: **NUT IPv4 Address Link 2** DestinationAddress: **TN2 IPv4 Address**

Payload (length: 8) Data: The Last 8 bytes of the IPv6/UDP Packet

Procedure:

0. Set IPv4 MTU on NUT to 1299. As a consequence the tunnel MTU should be 1280 (Minimum IPv6 MTU)
1. TN1 sends an “IPv6/UDP Packet” to the NUT for TN2bis with IPv6 Packet size = 1280 bytes.

Observable Results:

- **Step 1:** The NUT MUST fragment this Packet prior to encapsulate it and to forward it on IPv4 Network. It will result in 2 IPv4 Packets to receive on TN2 side: IPv6/UDP (1st Fragment) Tunneled and IPv6/UDP (Last Fragment) Tunneled. Their identifier is identical.

Test Sequence:

Tester	Link1 [IPv6]	RUT	Link2 [IPv4] MTU == 1299	Tester(IPv6)
TN1	IPv6/UDP ---------->	1		
			IPv6/UDP Tunneled Fragment ---------->	TN2(TN2Bis)
		Step 1	IPv6/UDP Tunneled Fragment ---------->	TN2(TN2Bis)

Postambule:

A successful run of this test case should have only modified the NDP and ARP caches. Thus, to put the NUT in its initial state they have to be cleaned.

After the test execution, the caches entries will be the following:

ARP Cache	NDP Cache
TR	TN1

5.4.1.3 Encapsulation/Decapsulation to IPv4 MTU Limits (1)

Purpose:

Set the IPv4 MTU on NUT to 1300. As a consequence the tunnel MTU should be 1280. Check Encapsulation of Echo Request and Decapsulation of Echo Reply (both have 1280 bytes for the IPv6 Packet) Messages. IPv6 Echo Request is sent from one host located in the 6to4 Network to one IPv6 Native host.

References:

- **[RFC3056]** ***Section 3***
- **[RFC3056]** ***Section 5.3***
- **[RFC3056]** ***Section 4***
- **[RFC2893]** ***Section 3.2:***

"**if** (IPv4 path MTU - 20) is less than or equal to 1280

if packet is larger than 1280 bytes

Send IPv6 ICMP "packet too big" with MTU = 1280.

Drop packet.

else

Encapsulate but do not set the Don't Fragment flag in the IPv4 header. The resulting IPv4 packet might be fragmented by the IPv4 layer on the encapsulating node or by some router along the IPv4 path.

endif

else

if packet is larger than (IPv4 path MTU - 20)

Send IPv6 ICMP "packet too big" with

MTU = (IPv4 path MTU - 20).

Drop packet.

else

Encapsulate and set the Don't Fragment flag in the IPv4 header.

endif

endif"

Resource Requirement:

- Packet generator
- Monitor To capture Packets:

Test Requirement:

- The IPv4 MTU on NUT has to be set to 1300. As a consequence the tunnel MTU should be 1280. If the NUT is not able to modify the MTU, It may be done using an ICMP Destination Unreachable message sent from TR to NUT as defined in the PATH MTU Discovery specification [RFC1191].

Discussion:

Echo Request sent by TN1 to IPv6 Native addresses must be tunneled by the NUT to TN2. If the IPv4 MTU on NUT is set to 1300, Packet coming from TN1 whose length is less or equal to 1280 bytes are Encapsulated and forwarded by the NUT. Moreover, IPv6 packets coming from outside the 6to4 Network are correctly decapsulated if their size is less or equal to 1280 bytes (the IPv4 Header is not included in the calculation).

Packets:

- ICMPv6 Echo Request (length: 1280 bytes)

IPv6 Header (length: 40)

Version: 6 TrafficClass: (0) FlowLabel: (0) PayloadLength: 1240 NextHeader: 58 HopLimit : (64) SourceAddress: **TN1 Global IPv6 Address with 6to4 Prefix** DestinationAddress: **TN2Bis Global IPv6 Native Address**
ICMPv6 Echo Request (length: 1240) Type: 128 Code: 0 Checksum: To calculate Identifier: (812) SequenceNumber: (0) **Payload (length: 1232)** Data: 1232 bytes of data

- ICMPv6 Echo Request Tunneled (length: 1300 bytes)

IPv4 Header (length: 20) Version: 4 IHL: 5 TypeOfService: 0 TotalLength: 1300 Identifier: XXXXX Reserved: 0 DF: 0 MF: 0 FragmentOffset: 0 TTL: XXXXX Protocol: 41 HeaderChecksum: XXXXX SourceAddress: **NUT IPv4 Address Link 2** DestinationAddress: **TN2 IPv4 Address**
Same as ICMPv6 Echo Request (length: 1280) **(Except than the Hop Limit is one less. ie set to 63 in our case)**

Procedure:

0. Set NUT IPv4 MTU to 1300. As a consequence the tunnel MTU should be 1280.
1. TN1 sends an “ICMPv6 Echo Request” to the NUT for TN2bis (IPv6 Packet size is 1280 bytes)

Observable Results:

- **Step 1:** The NUT tunnels this “ICMPv6 Echo Request” to TN2 in an IPv4 Packet “ICMPv6 Echo Request tunneled”

Test Sequence:

Tester	Link1 [IPv6]	RUT	Link2 [IPv4] MTU == 1300	Tester(IPv6)
TN1	ICMPv6 Echo Request ---------->	1 Step 1	ICMPv6 Echo Request Tunneled ---------->	TN2(TN2Bis)

Postambule:

A successful run of this test case should have only modified the NDP and ARP caches. Thus, to put the NUT in its initial state they have to be cleaned.

After the test execution, the caches entries will be the following:

ARP Cache	NDP Cache
TR	TN1

5.4.1.4 Encapsulation/Decapsulation to IPv4 MTU Limits (2) **

Purpose:

Set the IPv4 MTU on NUT to 1301. As a consequence the tunnel MTU should be 1281. Check Encapsulation of Echo Request and Decapsulation of Echo Reply (both have 1281 bytes for the IPv6 Packet) Messages. IPv6 Echo Request is sent from one host located in the 6to4 Network to one IPv6 Native host.

If a static value of 1280 is generated for the tunnel MTU, this test should be skipped.

References:

- **[RFC3056]** ***Section 3***
- **[RFC3056]** ***Section 5.3***
- **[RFC3056]** ***Section 4***
- **[RFC2893]** ***Section 3.2:***

"**if** (IPv4 path MTU - 20) is less than or equal to 1280

if packet is larger than 1280 bytes

Send IPv6 ICMP "packet too big" with MTU = 1280.

Drop packet.

else

Encapsulate but do not set the Don't Fragment flag in the IPv4 header. The resulting IPv4 packet might be fragmented by the IPv4 layer on the encapsulating node or by some router along the IPv4 path.

endif

else

if packet is larger than (IPv4 path MTU - 20)

Send IPv6 ICMP "packet too big" with

MTU = (IPv4 path MTU - 20).

Drop packet.

else

Encapsulate and set the Don't Fragment flag in the IPv4 header.

endif

endif"

Resource Requirement:

- Packet generator
- Monitor To capture Packets:

Test Requirement:

- If a static value of 1280 is generated for the tunnel MTU, this test should be skipped.
- The IPv4 MTU on NUT has to be set to 1301. As a consequence the tunnel MTU should be 1281. If the NUT is not able to modify the MTU, It may be done using an ICMP Destination Unreachable message sent from TR to NUT as defined in the PATH MTU Discovery specification **[RFC1191].**

Discussion:

Echo Request sent by TN1 to IPv6 Native addresses must be tunneled by the NUT to TN2. If the IPv4 MTU on NUT is set to 1301, Packet coming from TN1 whose length is less or equal to 1281 bytes are Encapsulated and forwarded by the NUT. Moreover, IPv6 packets coming from outside the 6to4 Network are correctly decapsulated if their size is less or equal to 1281 bytes (The IPv4 Header is not included in the calculation).

Packets:

- ICMPv6 Echo Request (length: 1281 bytes)

IPv6 Header (length: 40) Version: 6 TrafficClass: (0) FlowLabel: (0) PayloadLength: 1241 NextHeader: 58 HopLimit : (64) SourceAddress: **TN1 Global IPv6 Address with 6to4 Prefix** DestinationAddress: **TN2Bis Global IPv6 Native Address**
ICMPv6 Echo Request (length: 1241) Type: 128 Code: 0 Checksum: To calculate Identifier: (813) SequenceNumber: (0) **Payload (length: 1233)** Data: 1233 bytes of data

- ICMPv6 Echo Request Tunneled (length: 1301 bytes)

IPv4 Header (length: 20) Version: 4 IHL: 5 TypeOfService: 0 TotalLength: 1301 Identifier: XXXXX Reserved: 0 DF: 0 MF: 0 FragmentOffset: 0 TTL: XXXXX Protocol: 41 HeaderChecksum: XXXXX SourceAddress: **NUT IPv4 Address Link 2** DestinationAddress: **TN2 IPv4 Address**
Same as ICMPv6 Echo Request (length: 1281) **(Except than the Hop Limit is one less. ie set to 63 in our case)**

Procedure:

0. Set NUT IPv4 MTU to 1301. As a consequence the tunnel MTU should be 1281.
1. TN1 sends an “ICMPv6 Echo Request” to the NUT for TN2Bis (IPv6 Packet size is 1281 bytes)

Observable Results:

- **Step 1:** The NUT tunnels this “ICMPv6 Echo Request” to TN2 in an IPv4 Packet “ICMPv6 Echo Request tunneled”

Test Sequence:

Tester	Link1 [IPv6]	RUT	Link2 [IPv4] MTU == 1301	Tester(IPv6)
	ICMPv6 Echo Request			
TN1	---------->	1		
			ICMPv6 Echo Request Tunneled	
		Step 1	---------->	TN2(TN2Bis)

Postambule:

A successful run of this test case should have only modified the NDP and ARP caches. Thus, to put the NUT in its initial state they have to be cleaned.

After the test execution, the caches entries will be the following:

ARP Cache	NDP Cache
TR	TN1

5.4.2 To IPv6 Network of NUT

5.4.2.1 Unfragmentation at IPv4 Level

Purpose:

Check that IPv4 Fragmented Packets are reassembled by the NUT before to be forwarded on the 6to4 Network.

References:

- **[RFC2893]** ***Section 3.6***

Resource Requirement:

- Packet generator
- Monitor To capture Packets:

Test Requirement:

NONE

Discussion:

When an IPv6/IPv4 host or a router receives an IPv4 datagram that is addressed to one of its own IPv4 address, and the value of the protocol field is 41, it reassembles if the packet if it is fragmented at the IPv4 level, then it removes the IPv4 header and submits the IPv6 datagram to its IPv6 layer code.

The decapsulating node MUST be capable of reassembling an IPv4 packet that is 1300 bytes (1280 bytes plus IPv4 header).

The NUT is the end-point of automatic tunnel in a 6to4 stream coming from TN2 to TN1. It involves that IPv4 Fragmented Packets are reassembled before to be forwarded on the 6to4 Network.

Packets:

- Fragmented IPv4/IPv6/UDP Packet (1st Fragment) (length: 76 bytes)

IPv4 Header (length: 20)

Version: 4
IHL: 5
TypeOfService: 0
TotalLength: 76
Identifier: (528)
Reserved: 0
DF: 1
MF: 1
FragmentOffset: 0
TTL: (64)
Protocol: 41
HeaderChecksum: To calculate
SourceAddress: **TN2 IPv4 Address**
DestinationAddress: **NUT IPv4 Address Link 2**

IPv6 Header (length: 40)

Version: 6
TrafficClass: (0)
FlowLabel: (0)
PayloadLength: 16
NextHeader: 17
HopLimit : (64)
SourceAddress: **TN2Bis Global IPv6 Native Address**
DestinationAddress: **TN1 Global IPv6 Address with 6to4 Prefix**

UDP (length: 16)

SourcePort: (1000)
DestinationPort: (1000)
Length: 16
Checksum: To calculate

Payload (length: 8)
Data: (`01234567 89abcdef`)

- Fragmented IPv4/IPv6/UDP Packet (2nd Fragment) (length: 44 bytes)

IPv4 Header (length: 20)

Version: 4
IHL: 5
TypeOfService: 0
TotalLength: 44
Identifier: (528 Same as the previous Fragmented Packet)
Reserved: 0
DF: 1
MF: 1
FragmentOffset: 7 (in 8-octet units)
TTL: (64)
Protocol: 41
HeaderChecksum: To calculate
SourceAddress: **TN2 IPv4 Address**
DestinationAddress: **NUT IPv4 Address Link 2**

Payload (length: 24)
Data: 24 Bytes of data

- Fragmented IPv4/IPv6/UDP Packet (Last Fragment) (length: 44 bytes)

IPv4 Header (length: 20)

Version: 4
IHL: 5
TypeOfService: 0
TotalLength: 44
Identifier: (528)
Reserved: 0
DF: 1
MF: 0
FragmentOffset: 10 (in 8-octet units)
TTL: (64)
Protocol: 41
HeaderChecksum: To calculate
SourceAddress: **TN2 IPv4 Address**
DestinationAddress: **NUT IPv4 Address Link 2**

Payload (length: 8)
Data: 8 Bytes of data

- IPv6/UDP (length: 102 bytes)

IPv6 Header (length: 40)

Version: 6 TrafficClass: (0) FlowLabel: (0) PayloadLength: 48 NextHeader: 17 HopLimit : (63) **(The Hop Limit is one less than in the corresponding IPv6/UDP Fragmented packet)** SourceAddress: **TN2Bis Global IPv6 Native Address** DestinationAddress: **TN1 Global IPv6 Address with 6to4 Prefix**
UDP (length: 48) SourcePort: (1000) DestinationPort: (1000) Length: 48 Checksum: XXXXX **Payload (length: 40)** Data: The contains of payload of all fragmented IPv4/IPv6 Fragmented Packets

Procedure:

1. TN2 sends to the NUT the first fragment of one IPv4/IPv6/UDP Packet.
2. TN2 sends to the NUT the second and the last fragment of one IPv4/IPv6/UDP Packet.

Observable Results:

- **Step 1:** The NUT waits for receiving all fragments, reassembles all the fragments, desencapsulates this resulting packet and forwards this “IPv6/UDP Packet” to TN1 .

Test Sequence:

Tester	Link1 [IPv6]	RUT	Link2 [IPv4]	Tester(IPv6)
		1	IPv4/IPv6/UDP 1st Fragment <----------	TN2(TN2Bis)
		2	IPv4/IPv6/UDP 2nd Fragment <----------	TN2(TN2Bis)
TN1	ICMPv6 Echo Request Tunneled <----------	Step 2		

Postambule:

A successful run of this test case should have only modified the NDP and ARP caches. Thus, to put the NUT in its initial state they have to be cleaned.

After the test execution, the caches entries will be the following:

ARP Cache	NDP Cache
TR	TN1

6 Annexes

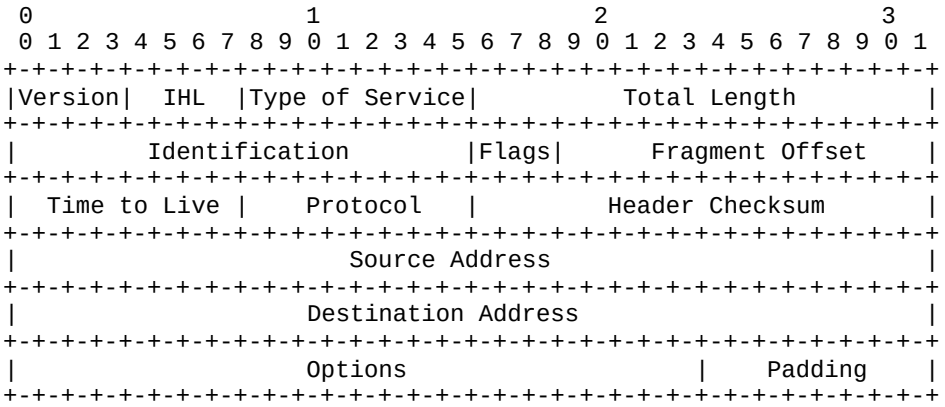
6.1 IPv4 packets Checksum computation

In this annexe we present how to calculate IPv4 related checksums of this test suite. It will be helpful for packets with the key-word "To Calculate".

6.1.1 IPv4 Checksum

The IPv4 Internet Header Format and the IPv4 Header Checksum calculation is defined in **[RFC791].**

The IPv4 Header Format is the following:

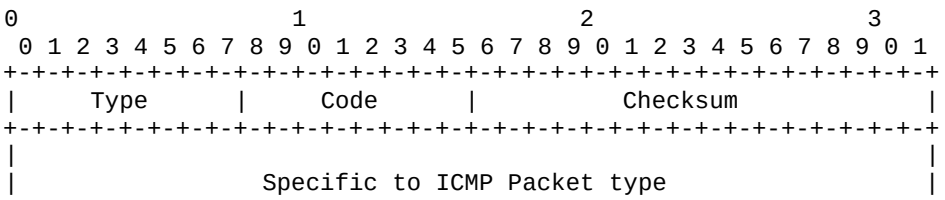


The different fields are explained in **[RFC791].** The checksum field is the 16 bit one's complement of the one's complement sum of all 16 bit words in the header. For purposes of computing the checksum, the value of the checksum field is zero.

6.1.2 ICMPv4 Checksum

The ICMPv4 Header Format and the ICMPv4 Header Checksum calculation is defined in **[RFC792].**

The ICMPv4 Header Format is the following:

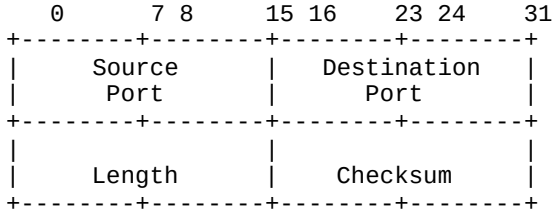


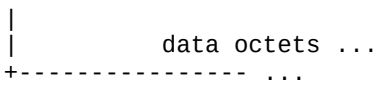
The different fields are explained in **[RFC792].** The checksum is the 16-bit ones's complement of the one's complement sum of the ICMP message starting with the ICMP Type.for computing the checksum , the checksum field should be zero. This checksum may be replaced in the future.

6.1.3 UDP Checksum

The UDP Header Format for IPv4 and the UDP Header Checksum calculation is defined in **[RFC768].**

The UDP Header Format is the following:



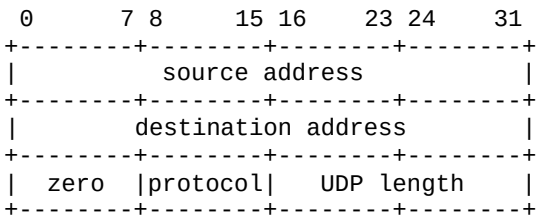


The different fields are explained in **[RFC768].**

Checksum is the 16-bit one's complement of the one's complement sum of a pseudo header of information from the IP header, the UDP header, and the data, padded with zero octets at the end (if necessary) to make a multiple of two octets.

The pseudo header conceptually prefixed to the UDP header contains the source address, the destination address, the protocol, and the UDP length. This information gives protection against misrouted datagrams.

This checksum procedure is the same as is used in TCP.

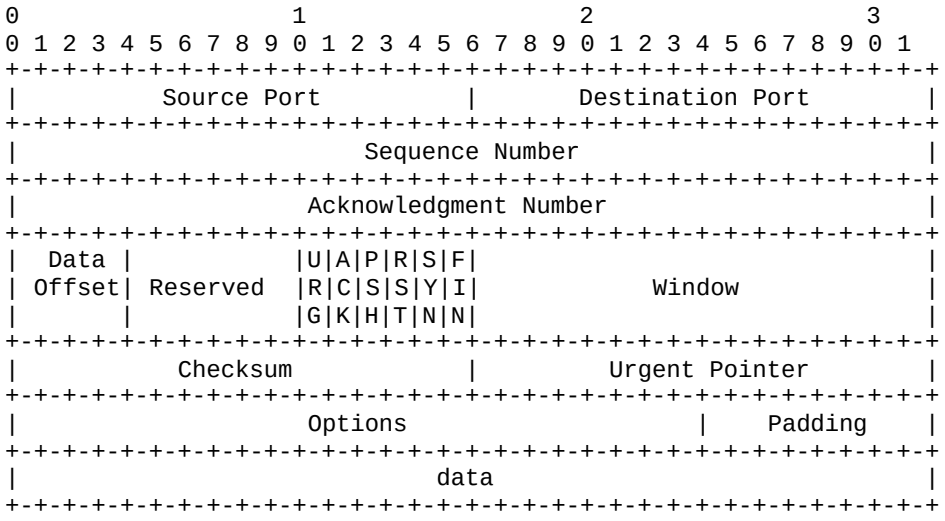


If the computed checksum is zero, it is transmitted as all ones (the equivalent in one's complement arithmetic). An all zero transmitted checksum value means that the transmitter generated no checksum for debugging or for higher level protocols that don't care).

6.1.4 TCP Checksum

The TCP Header Format for IPv4 and the TCP Header Checksum calculation is defined in **[RFC793].**

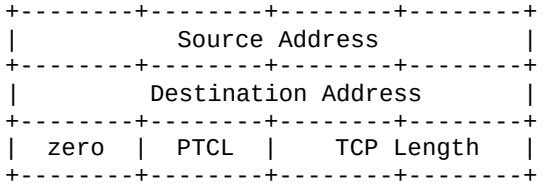
The TCP Header Format is the following:



The different fields are explained in **[RFC793].**

The checksum field is the 16 bit one's complement of the one's complement sum of all 16 bit words in the header and text. If a segment contains an odd number of header and text octets to be checksummed, the last octet is padded on the right with zeros toform a 16 bit word for checksum purposes. The pad is not transmitted as part of the segment. While computing the checksum, the checksum field itself is replaced with zeros.

The checksum also covers a 96 bit pseudo header conceptually prefixed to the TCP header. This pseudo header contains the Source Address, the Destination Address, the Protocol, and TCP length. This gives the TCP protection against misrouted segments. This information is carried in the Internet Protocol and is transferred across the TCP/Network interface in the arguments or results of calls by the TCP on the IP.



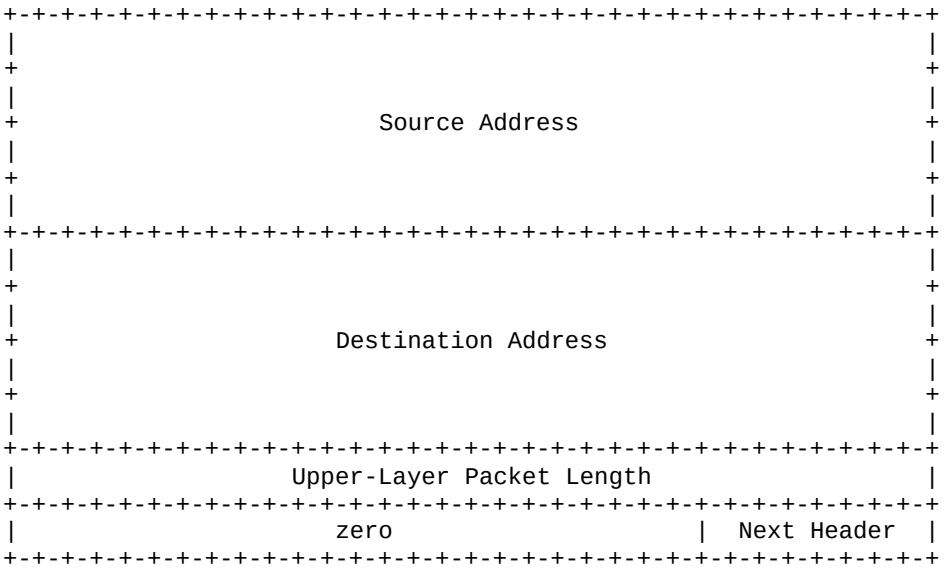
The TCP Length is the TCP header length plus the data length in octets (this is not an explicitly transmitted quantity, but is computed), and it does not count the 12 octets of the pseudo header.

6.2 IPv6 packets Checksum computation

In this annexe we present how to calculate IPv6 related checksums of this test suite. It will be helpful for packets with the key-word "To Calculate".

6.2.1 Pseudo-header

Any transport or other upper-layer protocol that includes the addresses from the IP header in its checksum computation must be modified for use over IPv6, to include the 128-bit IPv6 addresses instead of 32-bit IPv4 addresses. In particular, the following illustration **[RFC2460]** shows the TCP and UDP "pseudo-header" for IPv6:



If the IPv6 packet contains a Routing header, the Destination Address used in the pseudo-header is that of the final destination. At the originating node, that address will be in the last element of the Routing header; at the recipient(s), that address will be in the Destination Address field of the IPv6 header.

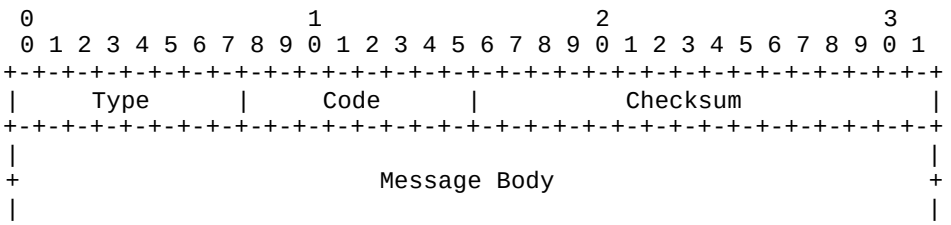
The Next Header value in the pseudo-header identifies the upper-layer protocol (e.g., 6 for TCP, or 17 for UDP). It will differ from the Next Header value in the IPv6 header if there are extension headers between the IPv6 header and the upper-layer header.

The Upper-Layer Packet Length in the pseudo-header is the length of the upper-layer header and data (e.g., TCP header plus TCP data). Some upper-layer protocols carry their own length information (e.g., the Length field in the UDP header); for such protocols, that is the length used in the pseudo-header. Other protocols (such as TCP) do not carry their own length information, in which case the length used in the pseudo-header is the Payload Length from the IPv6 header, minus the length of any extension headers present between the IPv6 header and the upper-layer header.

6.2.2 ICMPv6 Checksum

The ICMPv6 Header Format and the ICMPv6 Header Checksum calculation is defined in **[RFC2463].**

The ICMPv6 Header Format is the following:



The different fields are explained in **[RFC768].**

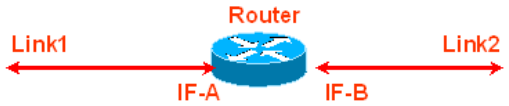
The checksum is the 16-bit one's complement of the one's complement sum of the entire ICMPv6 message starting with the ICMPv6 message type field, prepended with the previous "pseudo-header". The Next Header value used in the pseudo-header is 58.

6.2.3 UDP and TCP Checksums

The UDP/TCP Header Formats and checksums calculation are identical in IPv4 and IPv6 but IPv6 uses the previous pseudo-header in its checksum calculation

Unlike IPv4, when UDP packets are originated by an IPv6 node, the UDP checksum is not optional. That is, whenever originating a UDP packet, an IPv6 node must compute a UDP checksum over the packet and the pseudo-header.

6.3 Source Address Selection of ICMP Error Packets



Packet comes from IF-A and it should be forwarded to IF-B. If an ICMP Error Packet has to be generated by the router, the source address should be as described hereafter:

Error Case	I/F from which source address selected
No route	Any
Address unreachable	Any
Packet Too Big	IF-B
Time Exceed(Hop Limit)	Any

The keyword “Any” means every address involved in the routing process. Ie, one of the global addresses of IF-A, IF-B and Link-local address of IF-A (if the host is a neighbor).

Moreover, we consider that when an offending packet has to be put in the ICMP Error payload, this packet should be the one that arrived on IF-A although the Hop Limit may have already been decremented. A particular case is when the Hop Limit is 0 in the received packet. In this case an ICMP Time Exceeded packet has to be sent with the hop Limit set to 0 in the offending packet included.

7 REFERENCES

[NDThreats]
draft-ietf-send-psreq-03.txt, "IPv6 Neighbor Discovery trust models and threats"Nikander, P. (Ed.), April 2003, Internet Draft.

[6to4Security]
draft-ietf-v6ops-6to4-security-00.txt, P. Savola, October 2003, v6ops IETF Working Group, Internet Draft.

[RFC791]
RFC 791, STD0005, “Internet Protocol”, J. Postel, Sep-01-1981, STANDARD.

[RFC1191]
RFC 1191, Path MTU discovery, J.C. Mogul, S.E. Deering, Nov-01-1990, DRAFT STANDARD

[RFC1918]
RFC 1918, BCP0005, “Address Allocation for Private Internets”, Y. Rekhter, B. Moskowitz, D. Karrenberg, G. J. de Groot, E. Lear, February 1996, BEST CURRENT PRACTICE.

[RFC1981]
RFC 1981, “Path MTU Discovery for IP version 6”, J. McCann, S. Deering, J. Mogul, August 1996, DRAFT STANDARD [pub as:PROPOSED STANDARD]

[RFC2119]
BCP0014, “Key words for use in RFCs to Indicate Requirement Levels”, S. Bradner, March 1997, BEST CURRENT PRACTICE.

[RFC2460]
RFC 2460, "Internet Protocol, Version 6 (IPv6) Specification", Deering, S., and R. Hinden, December 1998, DRAFT STANDARD.

[RFC2461]
RFC 2461, "Neighbor Discovery for IP Version 6 (IPv6)", Narten, T., Nordmark, E. and W. Simpson, December 1998, DRAFT STANDARD.

[RFC2462]
RFC 2462, Thomson, S., and T. Narten, "IPv6 Stateless Address Autoconfiguration", December 1998, DRAFT STANDARD.

[RFC2463]
RFC 2463, "Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification", Conta, A., and S. Deering, December 1998, DRAFT STANDARD.

[RFC2464]
RFC 2464, “Transmission of IPv6 Packets over Ethernet Networks”, M. Crawford, December 1998, PROPOSED STANDARD.

[RFC2473]
RFC 2473, “Generic Packet Tunneling in IPv6 Specification”, Alex Conta and Stephen Deering, December 1998, PROPOSED STANDARD.

[RFC2529]
RFC 2529, “Transmission of IPv6 over IPv4 Domains without Explicit Tunnels”, B. Carpenter, C. Jung, March 1999, PROPOSED STANDARD.

[RFC2545]
RFC 2545, “Use of BGP-4 Multiprotocol Extensions for IPv6 Inter-Domain Routing”, P. Marques, F. Dupont, March 1999, PROPOSED STANDARD.

[RFC2893]
RFC 2893, “Transition Mechanisms for IPv6 Hosts and Routers”, R. Gilligan, E. Nordmark, August 2000, PROPOSED STANDARD.

[RFC3056]
RFC 3056, “Connection of IPv6 Domains via IPv4 Clouds”, B. Carpenter, K. Moore, February 2001, PROPOSED STANDARD.

[RFC3068]
RFC 3068, “An Anycast Prefix for 6to4 Relay Routers”, C. Huitema, June 2001, PROPOSED STANDARD.

[RFC3484]
RFC 3484, “Default Address Selection for Internet Protocol version 6 (IPv6)”, R. Draves, February 2003, PROPOSED STANDARD.

[RFC3513]
RFC 3513, “Internet Protocol Version 6 (IPv6) Addressing Architecture”, R. Hinden, S. Deering, April 2003, PROPOSED STANDARD.

[RFC3964]
RFC 3964, “ Security Considerations for 6to4”, P. Savola, C. Patel, December 2004, INFORMATIONAL